CORRECTED COPY. 27 APRIL 1967

CR 73085

A STUDY TO DETERMINE AN EFFICIENT DATA FORMAT AND DATA SYSTEM FOR A LIGHTWEIGHT DEEP SPACE PROBE

Final Report

31 October 1966

Prepared for

NASA AMES RESEARCH CENTER

Moffett Field, California

Under Contract No. NAS2-3254

TRW SYSTEMS

A STUDY TO DETERMINE AN EFFICIENT DATA FORMAT AND DATA SYSTEM FOR A LIGHTWEIGHT DEEP SPACE PROBE

FINAL REPORT

06488-6019-T000

31 October 1966

Prepared for
NASA Ames Research Center
Moffett Field, California
Under Contract NAS2-3254

TRW SYSTEMS
One Space Park · Redondo Beach, California

Prepared L. M. Bello
L. M. Bello

Approved A. Egger
A. Egger
Project Engineer

Approved R. E. Gottfried
R. E. Gottfried
Manager
Data Systems
Electronics Department

CONTENTS

CONTENTS (Continued)

CONTENTS (Continued)

ILLUSTRATIONS

ILLUSTRATIONS (Continued)

ILLUSTRATIONS (Continued)

TABLES

1. INTRODUCTION

This report summarizes the results obtained from a "Study to Determine an Efficient Data Format and Data System for a Lightweight Deep Space Probe" during the period 30 October 1965 to 30 October 1966. The study was conducted for NASA Ames Research Center (ARC) under contract NAS2-3254.

The study has resulted in the basic definition and overall design of a Central Data System (CDS) for a spin stabilized deep space probe which provides:

- Universal design for many missions
- One time development and qualification cost
- Capability of sampling a variable number of experiment inputs
- Capability to sample data (compress data to reduce bits required to transmit information)
- Capability to provide continuous data sampling during noncommunication periods (use of bulk store)
- Compatibility with the JPL DSIF.

The CDS design takes as its starting point the inputs from a group of eight solar experiments. These were chosen for the study by mutual agreement with NASA/ARC. Each experiment was studied in detail to determine the interface requirements in terms of the number of inputs to be sampled by the CDS, input signal characteristics,* required data processing, timing requirements, and sensor control signals.

Several solar orbital missions, ranging from 0.2 to 1.8 astronomical units (AU), were studied and their trajectories plotted to determine the communications link requirements. Curves were plotted to determine the

*Research Report No. 2, dated 29 August 1966.

possible transmission bit rates versus earth-spacecraft distance under specified telemetry link conditions, such as transmitting and receiving antenna gain and power, and error probability.

The CDS concepts were established based upon the results of studies which defined the experiment characteristics, probable mission profiles, and communication link requirements. The basic system concept developed is based upon sampling each data point at a high rate in a fixed sequence and allowing a central processor to select, process, and format sampled data into a highly efficient format for transmission. All data is processed by one central processor to eliminate redundant processing circuitry. This is in contrast to the Pioneer VI system developed by TRW Systems for NASA in which each experiment package performs its own data processing.

The CDS provides an input data sampler whose sampling is independent of transmission link requirements. That is, data is sampled at a fixed rate and temporarily stored for processing by the processor to provide continuous uninterrupted data flow to the spacecraft transmitter. The input data sampler samples maximum data each time a sampling cycle is initiated. The processor selects only the sampled data required to complete each frame according to a stored program and discards the rest. That is, more data is sampled than can be transmitted each spacecraft revolution.

The CDS provides for efficient formatting of sampled data. Sampled data is compressed and reduced by use of basic processor algorithms to maximize information content and minimize the required data rates. For example, there are no fill-in bits required within the CDS transmission formats. In contrast, fill-in bits are often required in a conventional data system, such as Pioneer VI, when the experiment performs processing at a rate asynchronous with the telemetry unit. Also as an example, log scaling can be used to achieve a reduction greater than 2 to 1 in the cosmic ray pulse data.

It is possible to extend the CDS processor capability to include certain forms of data reduction such as calculation of percent distribution, deviation from a mean, standard deviation, and possibly others to further

reduce the required transmission bit rate. This type of bit rate reduction is highly desirable, especially at low data transmission rates such as 64, 32, and 16 bps. A potential overall data compression of 10 to 1 may be achieved by the use of this type of data reduction.

The CDS processor provides a means for continually sampling the scientific environment even during periods of noncommunication with the ground station. Data is collected for up to 13 hours and is compressed and stored on magnetic tape for later transmission with realtime data, interleaving frames of realtime data with frames of stored data.

The programmable capabilities of the CDS also provide for desirable flexibility for a deep space probe operating within an unknown environment. That is, data formatting and processing may be reprogrammed in flight to optimize data transmission for an unexpected condition. For example, more data bits can be allocated to a specific experiment if the data rates from that experiment should be greater than expected during a mission. Also, the processor can be reprogrammed to bypass failed experiment packages.

The processor can also be used to evaluate spacecraft performance during the mission by periodically calling for a self test routine. The results of these test routines can either be individually transmitted to the ground within a special format, or a single test or word can be transmitted within the subcommutator after successfully completing the test routines.

A summary of the unique features of the CDS is as follows:

1) Each data point is sampled at a fixed, high rate and is processed and formatted by a programmable central processor.

2) The CDS utilizes a standard input signal interface.

3) The input data sampler is expandable up to 64 data points in a given sector.

4) The CDS has five programmable modes of operation.

5) The CDS has three fixed modes of operation which are not under program control.

6) The CDS can store data during periods of noncommunication for later transmission.

7) A command system is provided for uplink transmission of discrete commands and processor program instructions. All transmitted commands are self-verified prior to execution.

8) Convolutional coding* is used to improve transmission capability.

*Convolutional coding scheme developed by Dr. Dale R. Lumb, NASA/ARC.

2. MISSION AND DATA REQUIREMENTS

2.1 MISSION TRAJECTORIES AND DATA CHARACTERISTICS

To determine basic system requirements, trajectories were studied in terms of earth-spacecraft distance and spacecraft orbital velocity for various solar missions ranging from 0.2 AU to 1.8 AU. As a result of these studies, eight basic transmission bit rates were established to provide continuous communication coverage up to approximately 2 AU. These bit rates are 2048, 1024, 512, 256, 128, 64, 32, and 16 bps. Figure 2-1 shows a plot of transmission bit rates versus spacecraft distance from Earth based upon:

- 85-foot DSIF antenna
- 10^{-3} error rate
- 10-watt spacecraft transmitter
- 22-db spacecraft antenna
- Curve (a) no coding
- Curve (b) simple parity (parity every 8 bits)
- Curve (c) convolutional coding

Table 2-1 lists the values for determining the downlink communication capability at a 64-bps transmission rate. If the 210-foot dish antenna is used, transmission bit rates may be increased by a factor of 5, based on a 62-db gain for the 210-foot antenna.

Uplink (command) capabilities are shown in Figure 2-2 for an error rate of 10^{-6}. For a 2.0 AU distance, the command rate is 410 bps. This rate is well within the requirements of the CDS. Tables 2-2 and 2-3 list the values for determining the uplink communication capability at a transmission bit rate of 1K bps and 100 bps at 1.0 AU distances.

The required vehicle orbital velocity, with respect to the sun, versus spacecraft spin rate was studied to determine distance traveled between data samples. Studies have indicated that for a 0.2 AU mission, for example, the maximum spacecraft velocity is 283,000 ft/sec. For a spin rate of 1 rps, the spacecraft will travel a distance of 53.7 miles during each spacecraft revolution. This indicates that the distance between sampled data points is 53.7 miles when only one data point is taken per revolution. If a finer gradient measurement is required, multiple sensors may be used.

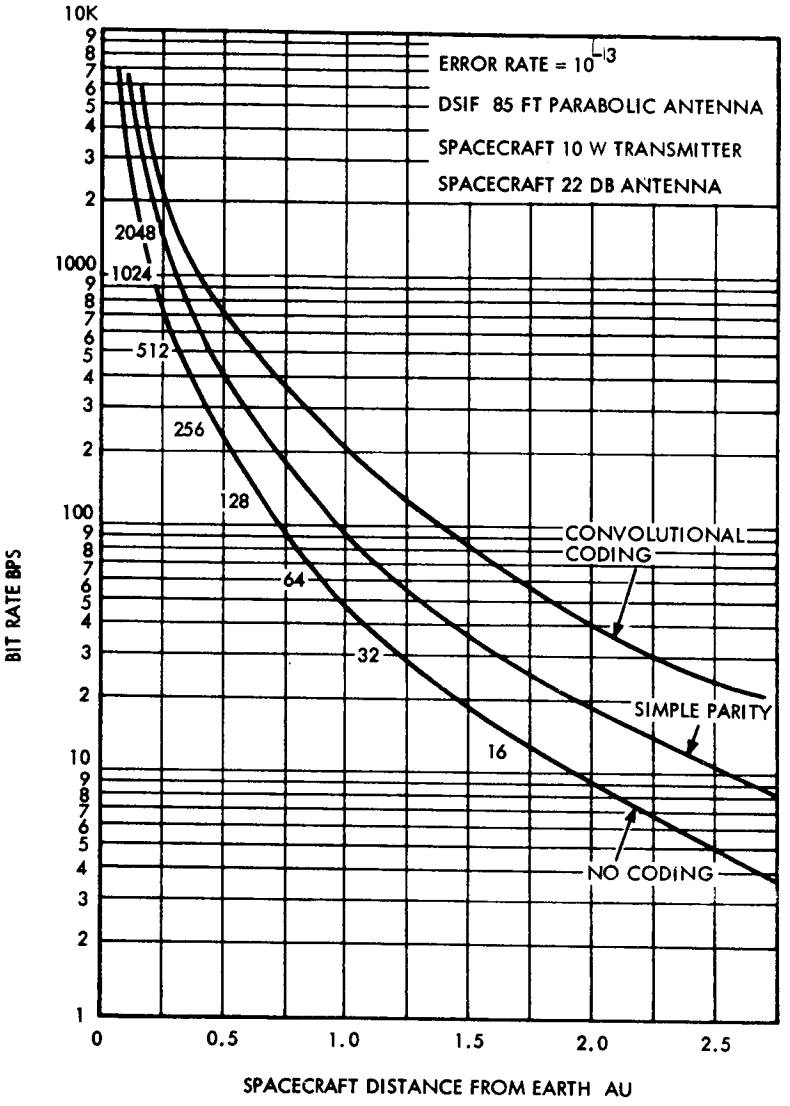


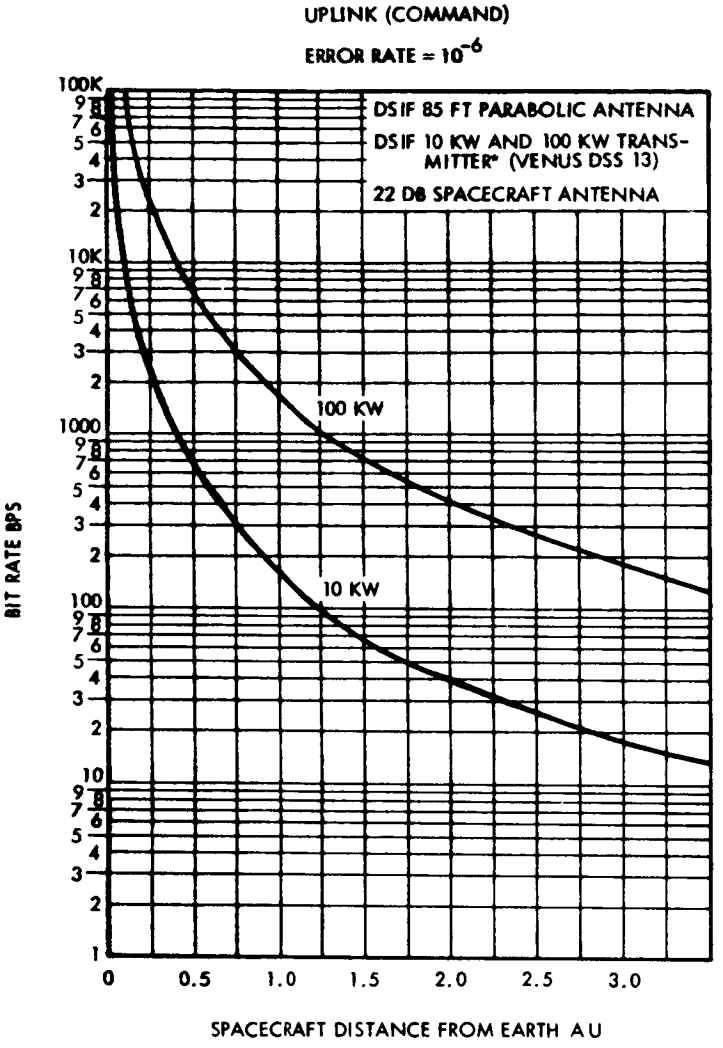
Figure 2-1. Downlink Bit Rate vs Distance

Table 2-1. Downlink Telecommunication Design Control
Coherent PSK, 64 bps

No.	Parameter	Value	Tolerance **		Source
			+	-	
1	Total transmitter power 10 w	40 dbm	-	-	-
2	Transmitting circuit loss	-1.6 db	0.5	0.65	PC-046
3	Transmitting antenna gain	22 db	-	-	-
4	Space loss, at 2292 MHz, $R = 95 \times 10^6$ KM	-259.2 db	0.0	0.0	-
5	Polarization loss	-3 db	0.3	0.3	PC-046
6	Receiving antenna gain	53.0 db	1.0	0.5	PC-046
7	Receiving circuit loss	-0.2 db	0.1	0.1	PC-046
8	Total received power	-149.0 dbm	1.9	1.55	-
9	Receiver noise spectral density T system = 55 $\pm$ 10°K	-181.2 dbm	0.9	0.7	PC-046
10	Carrier performance - Carrier modulation loss $\Delta\phi = 1.25 \pm 5\%$ radians	-10.0 db	1.4	1.8	-
11	Received carrier power	-159.0 db	3.3	3.35	-
12	Carrier APC noise BW ($2B_{LO} = 12$ Hz $^{+0.0}_{-10\%}$)	10.8 db Hz	0.5	0.0	PC-046
13	Threshold SNR in $2B_{LO}$	6 db	-	-	PC-046
14	Threshold carrier power	-164.4 dbm	1.4	0.7	-
15	Performance margin	5.4 db	4.7	4.05	-
16	Data channel - modulation loss	-0.5 db	0.2	0.2	-
17	Received subcarrier power	-149.5 dbm	2.1	1.75	-
18	Bit rate (1/T) 64 bps	18.1 db	0.0	0.0	-
19	Required ST/N/B ($Pe = 1 \times 10^{-3}$)	8.8 db*	-	-	-
20	Receiver losses and degradation	-2.0 db	0.0	1.0	PC-046
21	Threshold subcarrier power	-152.3 dbm	0.9	1.7	-
22	Performance margin	2.8 db	3.0	3.45	-

*Theoretical ST/N/B for $Pe = 1 \times 10^{-3}$ is 6.8 db plus 2 db detector and synchronizer degradation.

** Positive (+) tolerances result in an increase in SNR, negative tolerances result in a decrease in SNR.



* TABLES 2-2 AND 2-3 ARE BASED UPON THE USE OF THE 100-KW COMMAND TRANSMITTER AT THE GOLDSTONE VENUS TRACKING FACILITY AND THE 10 KW COMMAND TRANSMITTER AVAILABLE AT ALL OTHER DSS FACILITIES. SINCE LOADING OF THE CDS PROGRAM MEMORY IS THE MOST DEMANDING UPLINK REQUIREMENT, REQUIRING THE TRANSMISSION OF UP TO 18,000 BITS, IT IS DESIRABLE TO USE THE 100-KW FACILITY AT GOLDSTONE TO MAXIMIZE CDS REPROGRAMMING EFFICIENCY. NORMAL COMMAND ACTIVITY, ON THE OTHER HAND, CAN BE EFFICIENTLY ACCOMPLISHED BY ANY OF THE DSIF FACILITIES USING THEIR 10-KW COMMAND TRANSMITTER WITH CORRESPONDINGLY LOWER BIT RATES AS SHOWN BY THE 10-KW CURVE OF FIGURE 2-2. THE MODULATION INDEX FOR THE 10-KW TRANSMITTER WILL REMAIN 1.25 RADIANS AT 2 AU.

Figure 2-2. Uplink Bit Rate vs Distance

Table 2-2. Uplink Telecommunication Design Control Noncoherent FSK, Square Wave Subcarrier (100 KW Transmitter Power at 1K bps)

No.	Parameter	Value	Tolerance *		Source
			+	-	
1	Total transmitter power	80 dbm	-	-	-
2	Transmitting circuit loss	-0.4 db	0.1	0.1	PC-2.01
3	Transmitting antenna gain	51.0 db	1.0	0.5	PC-2.01
4	Space loss - at 2115 MHz, R = 1 AU	-262.5 db	0.0	0.0	-
5	Polarization loss	-3.0 db	0.5	0.5	-
6	Receiving antenna gain	21.5 db	-	-	-
7	Receiving circuit loss	-1.5 db	0.2	0.2	-
8	Total received power	-114.9 dbm	1.8	1.3	-
9	Receiver noise spectral density, T system = $2900^{\circ}K$	-164.0 db/Hz	1.0	0.5	-
10	Carrier performance - Carrier modulation loss $\Delta\phi = 1.25 \pm 5\%$ radians	-10.0 db	1.4	1.8	-
11	Received carrier power	-124.9 dbm	3.2	3.1	-
12	Carrier APC noise BW ($2B_{LO}$ = 20 Hz) $\pm$ 5%	13.0 db Hz	0.2	0.2	-
13	Threshold SNR in $2B_{LO}$	6.0 db	0.0	0.0	-
14	Threshold carrier power	-145.0 dbm	1.2	0.7	-
15	Performance margin	20.1 db	4.4	3.8	-
16	Data channel - modulation loss	-0.5 db	0.2	0.2	-
17	Received subcarrier power	-115.4 dbm	2.0	1.5	-
18	Bit rate (1/T) 1K bps	30.0 db	0.0	0.0	-
19	Required ST/N/B ($Pe = 1 \times 10^{-5}$)	16.4 db	1.0	1.0	-
20	Receiver losses and degradation	0.0	0.0	0.0	-
21	Threshold subcarrier power	-117.6 dbm	2.0	1.5	-
22	Performance margin	2.2 db	4.0	3.0	-

* Positive (+) tolerances result in an increase in SNR, negative tolerances result in a decrease in SNR.

Table 2-3. Uplink Telecommunication Design Control Noncoherent FSK, Square Wave Subcarrier (10 KW Transmitter Power at 100 bps)

No.	Parameter	Value	Tolerance *		Source
			+	-	
1	Total transmitter power	70 dbm	-	-	-
2	Transmitting circuit loss	-0.4 db	0.1	0.1	PC-2.01
3	Transmitting antenna gain	51.0 db	1.0	0.5	PC-2.01
4	Space loss - at 2115 MHz, R = 1 AU	-262.5 db	0.0	0.0	-
5	Polarization loss	-3.0 db	0.5	0.5	-
6	Receiving antenna gain	21.5 db	-	-	-
7	Receiving circuit loss	-1.5 db	0.2	0.2	-
8	Total received power	-124.9 dbm	1.8	1.3	-
9	Receiver noise spectral density, T system = 2900°K	-164.0 db/Hz	1.0	0.5	-
10	Carrier performance - Carrier modulation loss $\Delta\phi$ = 1.25 ± 5% radians	-10.0 db	1.4	1.8	-
11	Received carrier power	-134.9 dbm	3.2	3.1	-
12	Carrier APC noise BW ($2 B_{LO}$ = 20 Hz) ± 5%	13.0 db Hz	0.2	0.2	-
13	Threshold SNR in $2 B_{LO}$	6.0 db	0.0	0.0	-
14	Threshold carrier power	-145.0 dbm	1.2	0.7	-
15	Performance margin	10.1 db	4.4	3.8	-
16	Data channel - modulation loss	-0.5 db	0.2	0.2	-
17	Received subcarrier power	-125.4 dbm	2.0	1.5	-
18	Bit rate (1/T) 100 bps	20.0 db	0.0	0.0	-
19	Required ST/N/B ($Pe = 1 \times 10^{-5}$)	16.4 db	1.0	1.0	-
20	Receiver losses and degradation	0.0	0.0	0.0	-
21	Threshold subcarrier power	-127.6 dbm	2.0	1.5	-
22	Performance margin	2.2 db	4.0	3.0	-

*Positive (+) tolerances result in an increase in SNR, negative tolerances result in a decrease in SNR.

2.2 CDS EXPERIMENT GROUP

Consideration of the objectives of deep-space solar probe missions resulted in the choice of the following experiments for a spin-stabilized spacecraft weighing a maximum of 150 pounds. The choice of experiments was arrived at by mutual discussion and agreement with NASA/ARC.

1) Micrometeoroid. An experiment devised by the cosmic dust section of the Goddard Space Flight Center for measuring the level and variations in flux, momentum, and energy of minute particles in space.

2) Flux Gate Magnetometer. Based on an experiment devised at TRW for measuring the steady state and transient features of the magnetic field associated with the solar wind.

3) Cosmic Ray Detector. Using the basic concept developed by Dr. Simpson (University of Chicago) with minor changes in processing requirements to measure energy, composition, and distribution of protons, electrons, and alpha particles.

4) Plasma Probe. NASA/ARC experiment, devised by Dr. Wolfe. Used in Pioneer VI to provide detailed information on the protons and electrons of the interplanetary plasma.

5) Radio Propagation. The Stanford radio propagation experiment as designed for Pioneer VI.

6) Fast Neutron Detector. Based on the experiment devised at TRW for measuring the flux and energy spectrum of solar neutrons (1-20 mev range).

7) Very Low Frequency. A TRW experiment for investigating the mechanism which binds the collisionless solar wind into a fluid.

8) Cosmic Ray Experiment. Using the concept of Dr. Webber of the University of Minnesota except for minor changes in processing requirements to measure energy spectrum,

composition, and distribution of protons, alpha particles, and heavier nuclei.

The study indicates that all eight solar experiments are required for missions in the regions of 0.2 to 1.8 AU with the possible exception of the fast neutron detector experiment. The fast neutron experiment may be effective only between the 0.2 to 0.5 AU trajectory.

3. CENTRAL DATA SYSTEM GENERAL DESCRIPTION AND OPERATION

3.1 DESIGN CONCEPT

The CDS design is based upon the following basic concepts:

1) Sample input data at a fixed high rate and in a fixed sequence each sampling cycle regardless of mode of operation. Sample more raw data than can normally be transmitted each sampling cycle.

2) Store sampled data on a spacecraft revolution basis for later processing.

3) Process sampled data into a highly efficient format for transmission. Use a programmable processor to perform data compression, data reduction, and data formatting.

4) Provide system flexibility from mission to mission by utilizing standard experiment interfaces and programmable operation.

5) Provide the capability for in-flight program modification to optimize system operation to unexpected conditions.

6) Provide a bulk store unit to record sampled data during noncommunication periods, typically 13 hours each 24 hours.

7) Provide convolutional coding of CDS output to improve transmission capabilities and provide error detection and correction.

8) Provide a fixed realtime mode of operation which bypasses the processor during reprogramming.

3.2 BASIC CDS ELEMENTS

The basic CDS design consists of:

- Input data sampler
- Input data buffer
- Processor
- Output buffer
- Bulk store unit

- Fixed realtime transmission logic
- Convolutional coding logic
- Command processor.

A simplified block diagram of the CDS is shown in Figure 3-1. Data flow is shown in solid lines and control signals in dashed lines.

3.3 CDS CAPABILITIES

The CDS samples input data from experiment packages and from spacecraft performance sensors in the form of three basic types of signals (analog, pulse, and bi-level data). All analog data is encoded to 8-bit accuracy by one time-shared analog-to-digital (A/D) converter.

Each synchronous data source (depending upon spacecraft rotation) is sampled 64 times per spacecraft revolution and stored for processing by the central processor. The processor selects the desired samples from the input data buffer (IDB), performs the required data compression (i.e., log scaling, minimum-maximum determination, averaging, etc.), and formats the processed data for transmission. All processor operations are controlled by a stored program. The size of the output format is programmable to permit efficient utilization of the selected data bit rates.

The CDS timing and synchronization has been designed to provide an input data sampler (IDS) whose operation is essentially independent of transmission link requirements. That is, experiment data is sampled by the IDS and stored within the IDB until the processor is ready to select data for processing. The processed data is formatted and placed into one of the two output buffers for transmission. The only timing requirement is that data must be processed at a high enough rate to insure that sufficient data will be available for continuous uninterrupted transmission.

This independence of data sampling to transmission link requirements permits variable data processing to be performed prior to transmission. For example, as the allowable transmission bit rate decreases, due to the earth-spacecraft distance, the processor will select fewer samples to be transmitted per spacecraft revolution. Also more data compression will be performed on data that is transmitted, thus maximizing information transmitted with the available transmission bit rate.

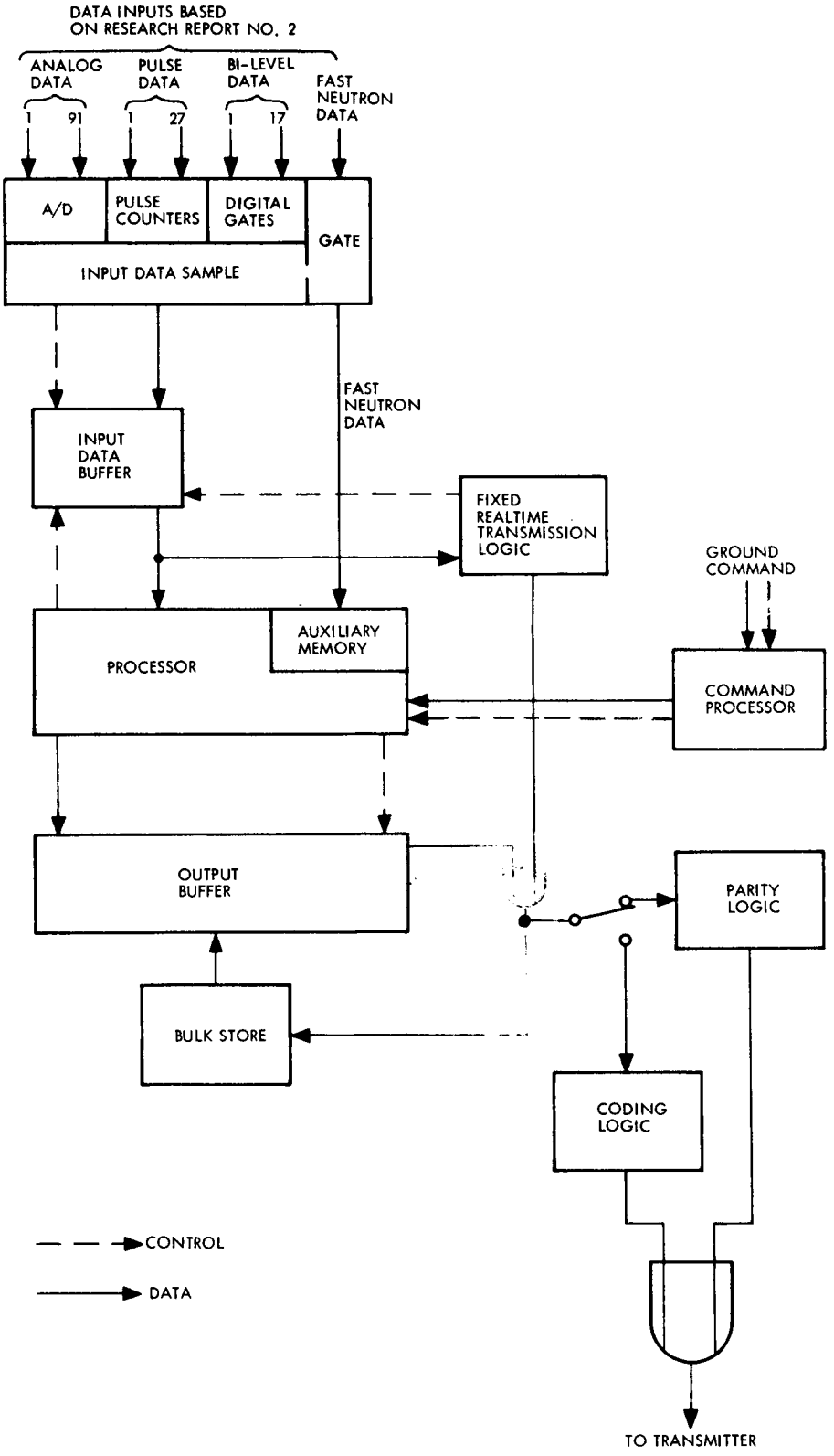


Figure 3-1. Simplified CDS Block Diagram

Figure 3-2 illustrates the types of input signals required for each experiment presented within Research Report No. 2. A summary of these signals together with the required control signals is given in Table 3-1. Table 3-1 indicates that A/D conversion is a requirement for all experiments. Thus, one A/D converter within the CDS may be used to perform encoding for all experiments thus eliminating circuit redundancy.

The following ground rules were used as a guide for establishing the input characteristics of the CDS.

1) The CDS will accept the following types of signals from the experiment packages:

   - Analog data in the range of 0 to +5 v
   - Bi-level data (0 and 5 v)
   - Pulse data for counting (0 and 5 v).

2) The experiments will perform all unique signal conditioning such as amplification, phase detection, threshold detection, etc.

3) The experiments will perform all unique gating that is peculiar to the individual experiment.

4) Where pulse height analysis is required, the experiment will sample the peak value and hold the value for measurement by the A/D converter. The CDS will then reset these sample and hold circuits for the next reading.

5) Pulse rates less than 10^5 pps and pulse widths greater than 2 μsec.

The CDS has five programmable modes of operation as shown in Figure 3-3. Realtime data interleaved with stored data refers to the operation where the spacecraft is in communication with the DSIF stations. This is expected to be for approximately 11 hours a day. During this time, the CDS will alternately transmit one frame of realtime data and one frame of stored data from the bulk store unit. The realtime data only mode consists of transmitting only realtime data (no bulk store data). This mode provides twice the realtime data rate of the realtime bulk store mode. The data storage mode refers to storing data into

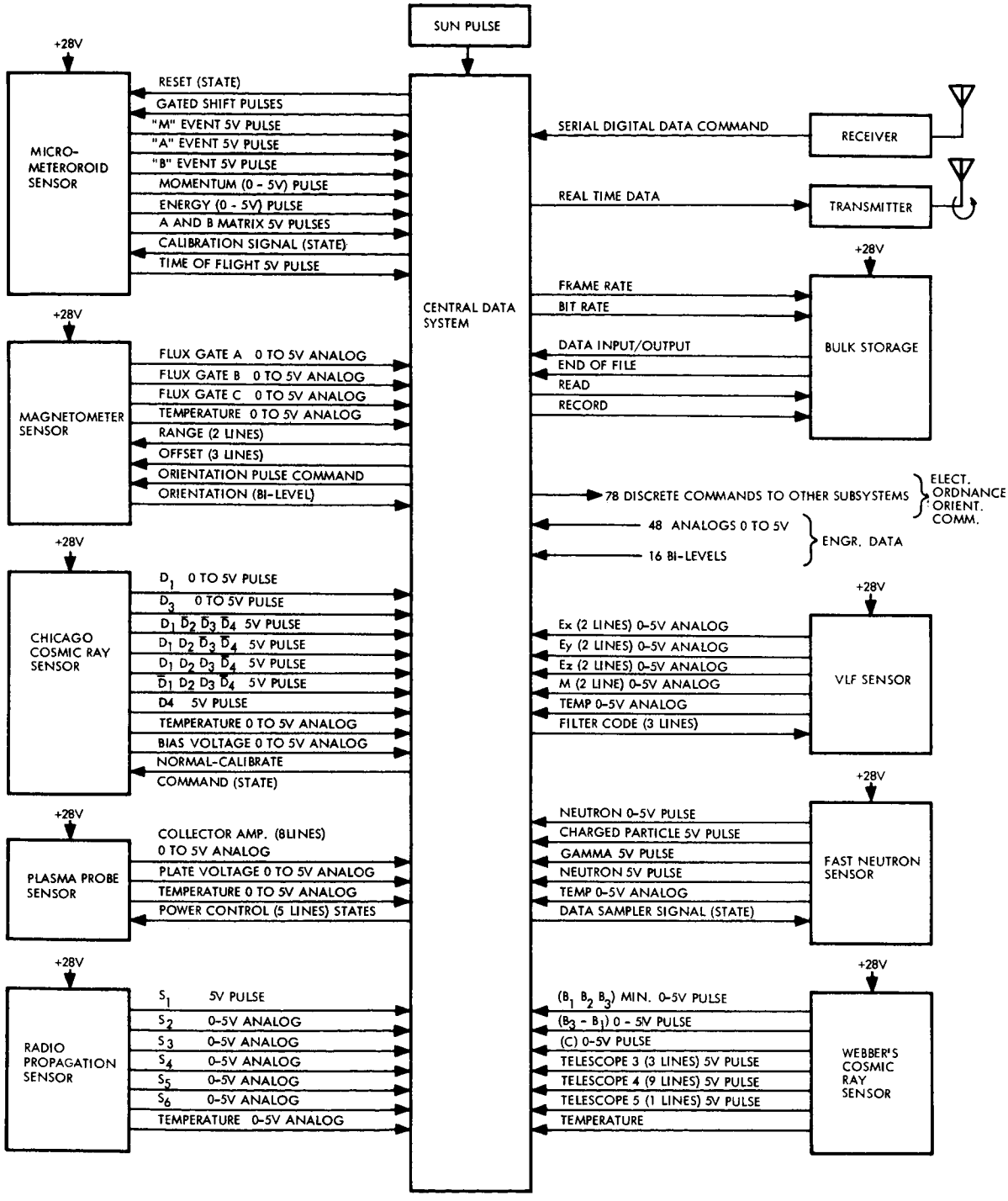


Figure 3-2. Detail CDS Electrical Interface

Table 3-1. Table of Data and Control Interface Signals

Source	Number of Data Signals				Number of Control Signals	
	5-volt Pulse for Counting	0 to 5-volt Pulse for PHA*	0 to 5-volt Analog for A/D	Discretes (bi-levels)	States	Pulses
Micrometeoroid	5	2	-	-	2	1
Magnetometer	-	-	4	1	5	1
Chicago Cosmic Ray	5	2	2	-	1	-
Plasma Probe	-	-	10	-	5	-
Radio Propagation	1	-	6	-	-	-
Fast Neutron	3	1	1	-	1	-
VLF	-	-	9	-	3	-
Webber's Cosmic Ray	13	3	1	-	1	-
Housekeeping	-	-	48	16	-	-
TOTALS	27	8	83	17	18	2

*PHA = pulse height analysis requires A/D conversion.

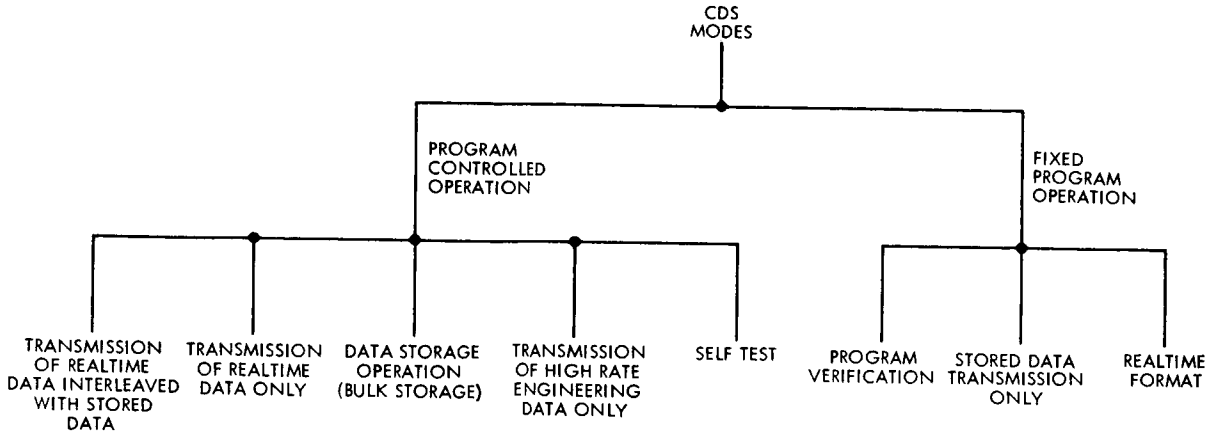


Figure 3-3. CDS Modes of Operation

the bulk store; this mode is expected to be required approximately 13 hours per day. High rate engineering only refers to the transmission of spacecraft performance and experiment status data at a high rate during launch phases or during times of suspected spacecraft malfunction. A self test mode is provided to evaluate CDS performance during the course of the mission.

Each programmable mode requires an individual program; however, common subroutines are shared among all programs. That is, the processor program memory is organized with one section or bank containing subroutines and the other bank containing the basic programs. Whenever a basic program requires a subroutine, the basic program branches off into the subroutine. After performing the subroutine a jump is made back to the basic program.

The three fixed operational modes are designed to bypass the processor and are used to transmit a limited quantity of scientific and engineering data during periods when the processor program is being modified and during times when a processor malfunction is suspected. The input data sampler operates in its normal sequence during this mode of operation. However, instead of the processor selecting data samples from the IDB, samples are selected from the IDB by a fixed programmer. The selected data is directly formatted by the use of a combiner and is fed directly to either the block coding circuitry or directly to the modulator.*

3.3.1 Input Data Sampling

The basic requirement of the IDS is to:

- Sample analog, bi-level, and pulse data
- Analog to digital encode inputs when required
- Store this data into the IDP for later processing by the processor.

*For the purpose of the study, the specification of the modulation scheme was not required. If a biphase modulation scheme similar to that used in the Pioneer VI is selected, this scheme can be simply implemented within the CDS (requires two or three integrated circuits).

The IDS as shown in Figure 3-4 consists of the following units:

- Master oscillator
- Spacecraft sector generator
- Fixed program counter
- Decoder
- Input gates
- Pulse counters
- A/D converter and redundant unit.

The clock provides the basic timing required for the sampler. The spacecraft sector generator divides each spacecraft revolution into 64 equal sectors, compensating for changes in spin rate. The fixed programmer provides the required counts or states which are decoded by the decoder which in turn enables the input gates at the proper time for sampling CDS inputs. The fixed programmer is synchronized by the spacecraft sector generator and output buffer marker signals. The marker signal is generated when transmission is completed from either of the two output buffers and is used to synchronize data sampling to data transmission. Bi-level data is sampled and stored directly into the IDB. Pulse data is accumulated within counters and then stored within the IDB. Analog inputs are sampled and encoded into an 8-bit digital number for entry into the IDB. Note that a redundant A/D converter is provided for increased reliability.

As stated earlier, each spacecraft revolution is divided into 64 equal segments called sectors. Each sector is also divided into 64 segments called words. The choice of 64 sectors per revolution is based upon the requirements of the plasma experiment. The angular resolution required is 5 5/8 degrees in the ecliptic plane. This resolution is achievable by dividing 360 degrees by 64 which equals 5 5/8 degrees. (Refer to Research Report No. 2 for more detailed characteristics of the plasma experiment.) The sector period is dependent upon spacecraft spin rate whereas each word period is determined and fixed by a crystal controlled oscillator. The characteristics of one spacecraft sector are shown

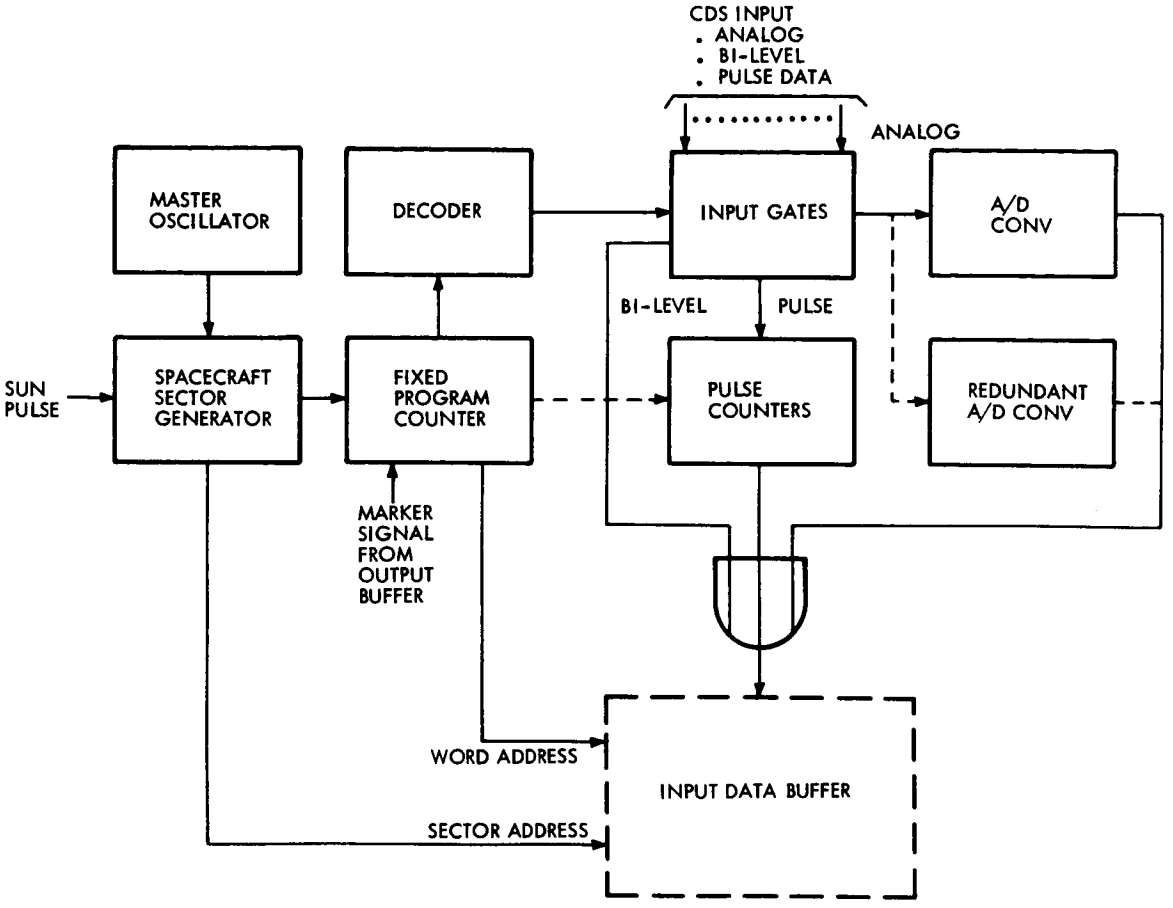


Figure 3-4. Input Data Sampler Block Diagram

in Figure 3-5. Analog inputs are sampled, encoded, and stored within the IDB during word times 1 through 20. Pulse data is sampled for one sector and stored within the IDB during word times 21 through 34.

The sampling technique utilized for sampling pulse data is shown in Figure 3-6. Pulse data is accumulated from the end of word 24 of sector 6 to the beginning of word 24 of sector 7. The accumulated counts are stored within the IDB during word 24. This sampling and storing sequence is repeated each sector. The next pulse data input, for example, would be sampled from the end of word 25 of sector 6 to the beginning of word 25 of sector 7. The accumulated counts from this pulse gate are stored within the IDB during word time 25, etc. If a counter requires more than 8 bits for example for the maximum allowable input rate of 10^5 pps, two word times are allotted for storing the accumulated data into the IDB.

This pulse data sampling technique provides for accumulating pulses during 63 out of 64 words of each sector. Assuming a 1-rps spin rate, each word period is 244 μsec and each sector period is 15.622 msec. Therefore, at the maximum allowable CDS input counting rate (10^5 pps) approximately 50 counts out of a possible 1562 are missed in one sector or 3.2 percent. This assumes two 8-bit words were required to accumulate the counts and two word times were required to transfer and store the accumulated counts within the IDB. Note that the majority of the pulse data only requires a one word counter and therefore only 1.6 percent of the counts are missed in one sector.

Since the sector period will vary with spin rate, the counting period for accumulating counts will also vary. However, since spin rate is transmitted within the subcommutator (subcomm) the sector period may be calculated.

As shown in Figure 3-5, words numbered 37 through 64 within each sector are spares. This provides the capability to increase the number of inputs for possible future expansion. However, the unused word times act as a safety margin should the spin rate increase. For example, if the spin rate should be greater than the expected 1 rps, the sector period decreases accordingly allowing less time to sample the required words. Since only 37 of the available 64 words are required to sample experiment

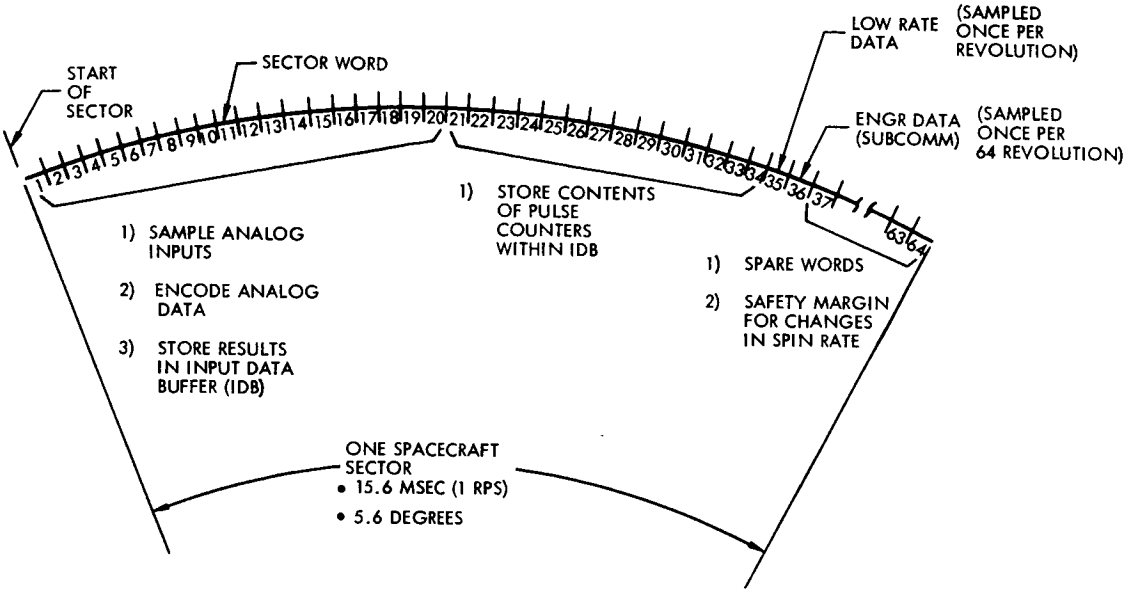


Figure 3-5. Spacecraft Sector Detail Diagram

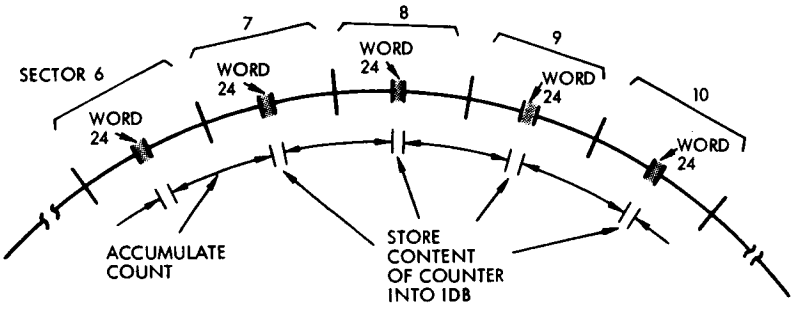


Figure 3-6. Typical Pulse Data Sampling for Word 24 (Webber's Cosmic, see Table 3-2)

data, the spin rate could increase up to 1.7 rps without adversly affecting the input data sampling. Note that increasing the number of words used per sector decreases the allowable increase in spin rate. A spin rate of less than the designed 1 rps has no effect upon operation of the IDS.

The IDB is addressed by the spacecraft sector generator along the Y axis and by the fixed program counter along the X axis. Figure 3-7 illustrates the basic organization of the IDB. Sampled data is stored in a fixed sequence within the IDB in order to correlate data with the sector address from which it was sampled. For example, data sampled in sector 1 is always stored in the sector 1 cell within the IDB. Thus when the processor selects data from this particular cell, it knows that the information contained within this cell was sampled from sector 1.

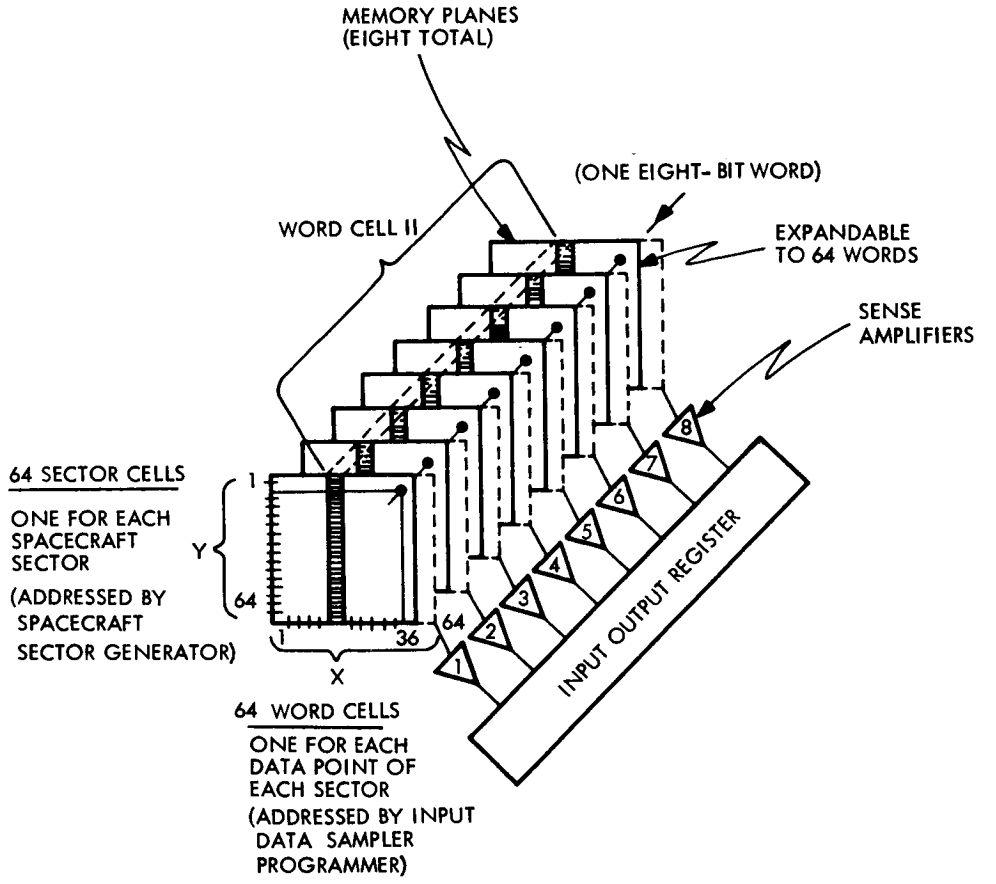
Table 3-2 lists each experiment data point and also specifies the IDB cell location assigned to each data point. The table is divided into three groups:

- High rate data
- Low rate data
- Subcomm data.

High rate data is defined as data that is sampled 64 times per spacecraft revolution or during each sector. Low rate data is sampled once per revolution. Subcomm data is sampled once per 64 revolutions.

Word cells 1 through 34 are used to store high rate data within the IDB. The word cells listed in the table indicate word cell address of the IDB (X axis) which correspond to the word numbers of each sector as shown in Figure 3-5. That is, data collected during word 11 is stored within word cell 11 of the IDB (X axis). Since high rate data is sampled 64 times per revolution, 64 data samples are stored within word cell 11 as indicated in Figure 3-7.

Low rate data is assigned word cell 35. Since each low rate data is sampled only once per revolution, all low rate data is stored within one IDB word cell. Each of the 22 low rate data words is stored within 22 sector cells of word cell 35 of the IDB. Note that 42 of the available 64 sector cells of word cell 35 are not used.



1) Sector cells refer to spacecraft sectors.
2) Word cells refer to words within each sector.
3) Words 37 through 64 are spare.

Figure 3-7. Input Data Buffer Organization

Table 3-2. IDB Word Cell Assignment

A. High Rate Data

Word Cell	Experiment	Signal Name	Signal Type	Bits
1	MAG	X AXIS	A	8
2	MAG	Y AXIS	A	8
3	MAG	Z AXIS	A	8
4	VLF	X AXIS	A	6 + (2 spares)
5	VLF	Y AXIS	A	6 + (2 spares)
6	VLF	Z AXIS	A	6 + (2 spares)
7	C. COS	D_1	PHA	7 + (1 spare)
8	C. COS	D_3	PHA	5 + (3 spares)
9	W. COS	(C)	PHA	8
10	W. COS	(B_1 B_2 B_3)	PHA	8
11	W. COS	(B_3 - B_1)	PHA	8
12	PLA	CH. 1	A	8
13	PLA	CH. 2	A	8
14	PLA	CH. 3	A	8
15	PLA	CH. 4	A	8
16	PLA	CH. 5	A	8
17	PLA	CH. 6	A	8
18	PLA	CH. 7	A	8
19	PLA	CH. 8	A	8
20	RAP	S_5	A	6 + (2 spares)
21	W. COS	TEL 3.1	P	5 + (3 spares)
		TEL 3.2	P	5 + (3 spares)
		TEL 3.3	P	5 + (3 spares
22	W. COS	TEL 4.1	P	8
23	W. COS	TEL 4.2	P	8
24	W. COS	TEL 4.3	P	5 + (3 spares)
25	W. COS	TEL 4.4	P	5 + (3 spares)
		TEL 4.5	P	5 + (3 spares)

Table 3-2 (Continued)

Word Cell	Experiment	Signal Name	Signal Type	Bits
26	W. COS	TEL 4.6	P	5 + (3 spares)
		TEL 4.7	P	5 + (3 spares)
		TEL 4.8	P	5 + (3 spares)
		TEL 4.9	P	5 + (3 spares)
27	W. COS	TEL 5 (1)	P	8
28	W. COS	TEL 5 (2)	P	4 + (4 spares)
29	C. COS	D_4 (1)	P	8
30	C. COS	D_4 (2)	P	3 + (5 spares)
31	C. COS	$D_1 \overline{D}_2 \overline{D}_4$	P	8
32	C. COS	$D_1 D_2 \overline{D}_3 \overline{D}_4$	P	4 + (4 spares)
33	C. COS	$D_1 D_2 D_3 \overline{D}_4$	P	4 + (4 spares)
34	C. COS	$D_1 D_2 D_3 \overline{D}_4$	P	2 + (6 spares)

B. Low Rate Data

Word Cell	Signal Name		Signal Type	Bits
35-1	Start time of scanning cycle (ET_1)		P	8
35-2	Start sector address		P	8
35-3	MIC, RAP, VLF ID, and experiment normally calibrate status		P	6 + (2 spares)
35-4	Plasma voltage and channel ID		P	8
35-5	VLF M_1		A	6 + (2 spares)
35-6	VLF M_2		A	6 + (2 spares)
35-7	VLF X_c	Correlated data	A	6 + (2 spares)
35-8	VLF Y_c	Correlated data	A	6 + (2 spares)
35-9	VLF Z_c	Correlated data	A	6 + (2 spares)
35-10	VLF filter		P	4 + (2 spares)
35-11	RAP S_1		P	7 + (2 spares)

Table 3-2 (Continued)

Word Cell	Signal Name	Signal Type	Bits
35-12	RAP S_2	A	6 + (2 spares)
35-13	RAP S_3	A	6 + (2 spares)
35-14	RAP S_4	A	6 + (2 spares)
35-15	RAP S_6	A	6 + (2 spares)
35-16	MIC A and B matrix	P	8
35-17	MIC energy	A	4 + (4 spares)
35-18	MIC MOM	A	4 + (4 spares)
35-19	MIC TOF	P	4 + (4 spares)
35-20	MIC sun aspect angle	P	7 + (1 spare)
35-21	A, B, and M event	P	3 + (5 spares)
35-22	MIC elapsed time	P	4 + (4 spares)
C. Subcommutated Data		A, Bi, P	8
36-64	Engineering data Bi-level ET_2 ωt MAG Range Offset Orientation		

NOTE: Refer to the glossary in Appendix A for definition of terms.

Subcomm data is stored within sector cell 64 of word cell 36. Since only one 8-bit subcomm data word is transmitted each spacecraft revolution, only one 8-bit word cell within the IDB is required to store data. Sector cell 64 of word cell 35 could have been used to store subcomm data since this word is not used. However, a separate word cell was used for the subcomm to simplify the sampling and storing of engineering data during the high rate engineering data mode.

3.3.2 Data Processing and Formatting

3.3.2.1 Data Processing

Data processing is defined as the operations performed on sampled data by the processor prior to transmission. Processing operations include:

- Selection of specific data from the IDB and from the auxiliary memory
- Data processing
- Data formatting.

These operations are programmable and are under the control of a processor instruction memory.

The processor consists of a data operator, an instruction memory, and an auxiliary memory (Figure 3-8).

The data operator performs all required operations on selected data samples under the control of the processor instruction memory. The data operator consists of two 14-bit auxiliary (AR2, AR3) registers, a 30-bit accumulator (AR1), and an adder-subtractor to perform the required arithmetic and logical operations. The auxiliary registers are working registers which store 8 bits of data and 6 bits of sector information.

The processor instruction memory is a word oriented, ferrite core system with a capacity of 18,432 bits. Each instruction word contains 18 bits.

The auxiliary memory is a word oriented ferrite core system with a capacity of approximately 30,000 bits. The auxiliary memory is primarily used for storage of data from the fast neutron experiment.

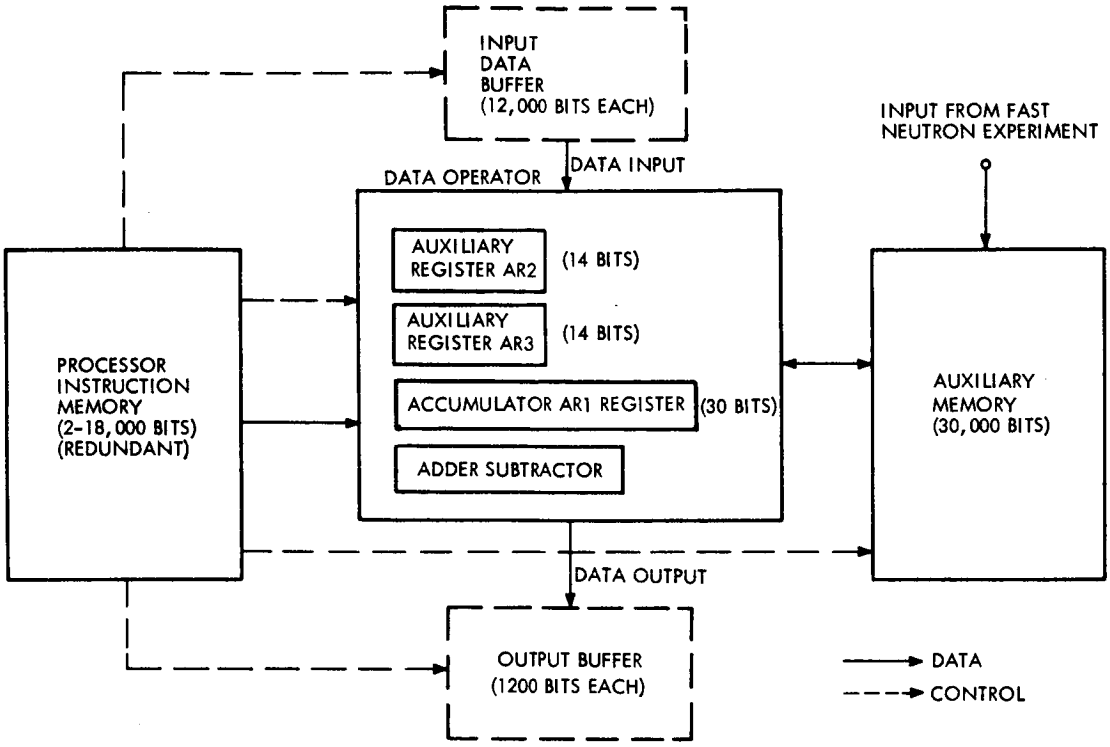


Figure 3-8. Processor Simplified Block Diagram

A small percentage of its capacity is used by the data operator as a temporary store for calculations such as determining average values of data samples collected for a period of 1 hour during conditions of solar activity, stored within the auxiliary memory, and transmitted at a low rate over a period of 20 hours.

The processor can perform specific data compression such as log scaling, averaging, determining minimum and maximum values of n samples, and any other data operation which can be performed by basic addition, subtraction, and shifting left and right. At lower bit rates, the processor reduces the number of bits per frame by selecting fewer samples per spacecraft revolution and by performing more data compression such as log scaling data, finding minimum-maximum values, etc. Refer to 4.2.9 for a discussion of processing algorithms.

Formatting of processed data is automatically performed in the transfer of data from the data operator to the output buffers. During the transfer operation the processor performs the following formatting steps:

1) Selects the frame synchronization bits from the processor instruction memory and enters them into the appropriate output buffer memory locations.

2) Enters subcomm, bit rate, and format identification (ID) into appropriate output buffer locations.

3) Stores processed data into pre-selected output buffer locations. Stores sector ID bits to identify data with its sector sampling address where required; i.e., minimum-maximum.

4) Determines the size of each frame according to the processor program.

5) Formats and stores each frame of data to permit continual uninterrupted realtime transmission of data alternately from the two output buffers.

A typical processor program flow diagram is shown in Figure 3-9. Note that the last block of the flow diagram says "await marker signal from output buffer." This is also a timing requirement similar to that used

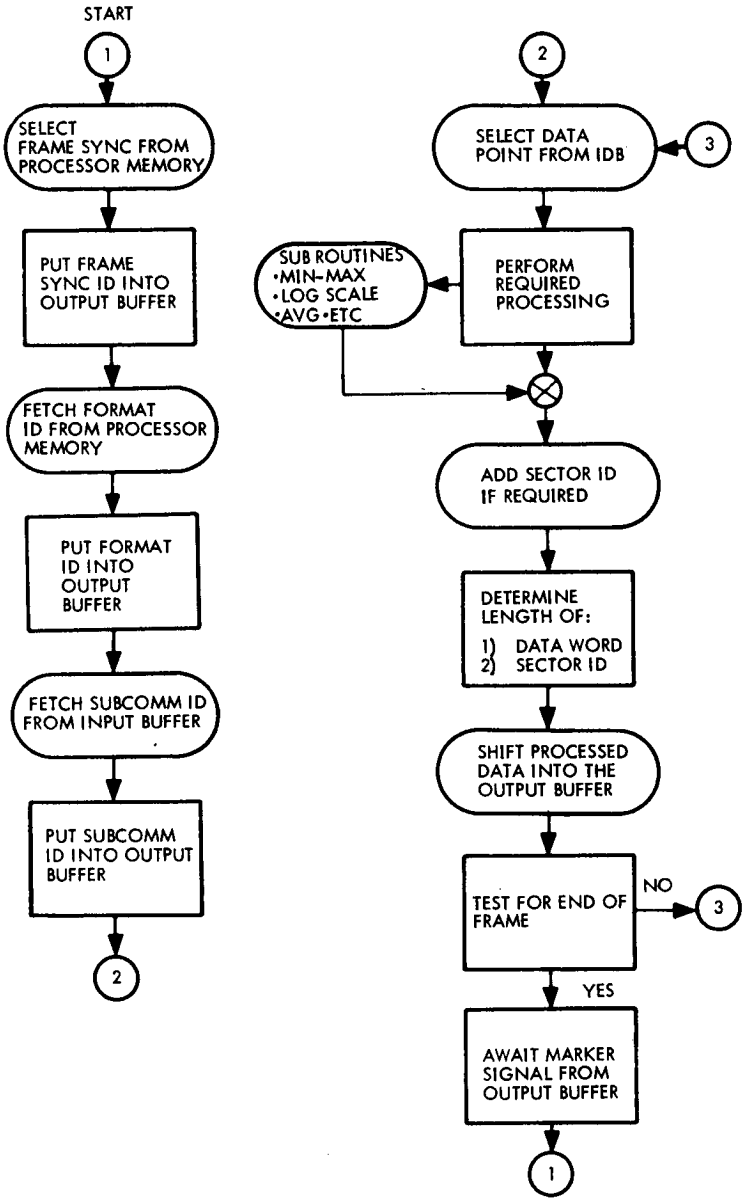


Figure 3-9. Flow Diagram of Processing and Formatting Operations

to initiate sampling by the IDS. This timing will insure that the output buffer has been completely read out before data is entered into it for the next cycle. In addition, a third timing requirement must be met. That is, transmission time for complete transmission of data from one output buffer, at the prevailing bit rate, must be longer than the time required to process and fill the other. This will insure that a complete frame or frames of data are ready for transmission prior to the time transmission begins from the output buffer.

3.3.2.2 Data Formats

A typical format for the 1024-bps transmission rate is shown in Figure 3-10. This is only one example of any number of formats which are possible by programming. The format is divided into four basic sections: frame identification, high rate data, low rate data, and subcommutated data. Each section is discussed below.

FRAME IDENTIFICATION

Frame Synchronization. Frame ID contains a 24-bit pseudo random code for frame synchronization.

Subcomm ID. Six bits are allocated as subcomm ID to identify each of the 64 subcomm words which are transmitted once per frame.

Format ID. Four bits are allocated to identify which of the 16 formats is being used and modes of operation (bulk store, realtime, etc.).

Bit Rate ID. Three bits identify each of the eight transmission bit rates.

HIGH RATE DATA This refers to experiment data that is sampled many times per revolution.*

Magnetometer. In this format, the X, Y, and Z axis magnetometer flux is encoded to 8-bit accuracy. Thus a total of 24 bits are required. Each sample is taken in regular known

*Refer to Research Report No. 2 for an explanation of experiment sampling experiments.

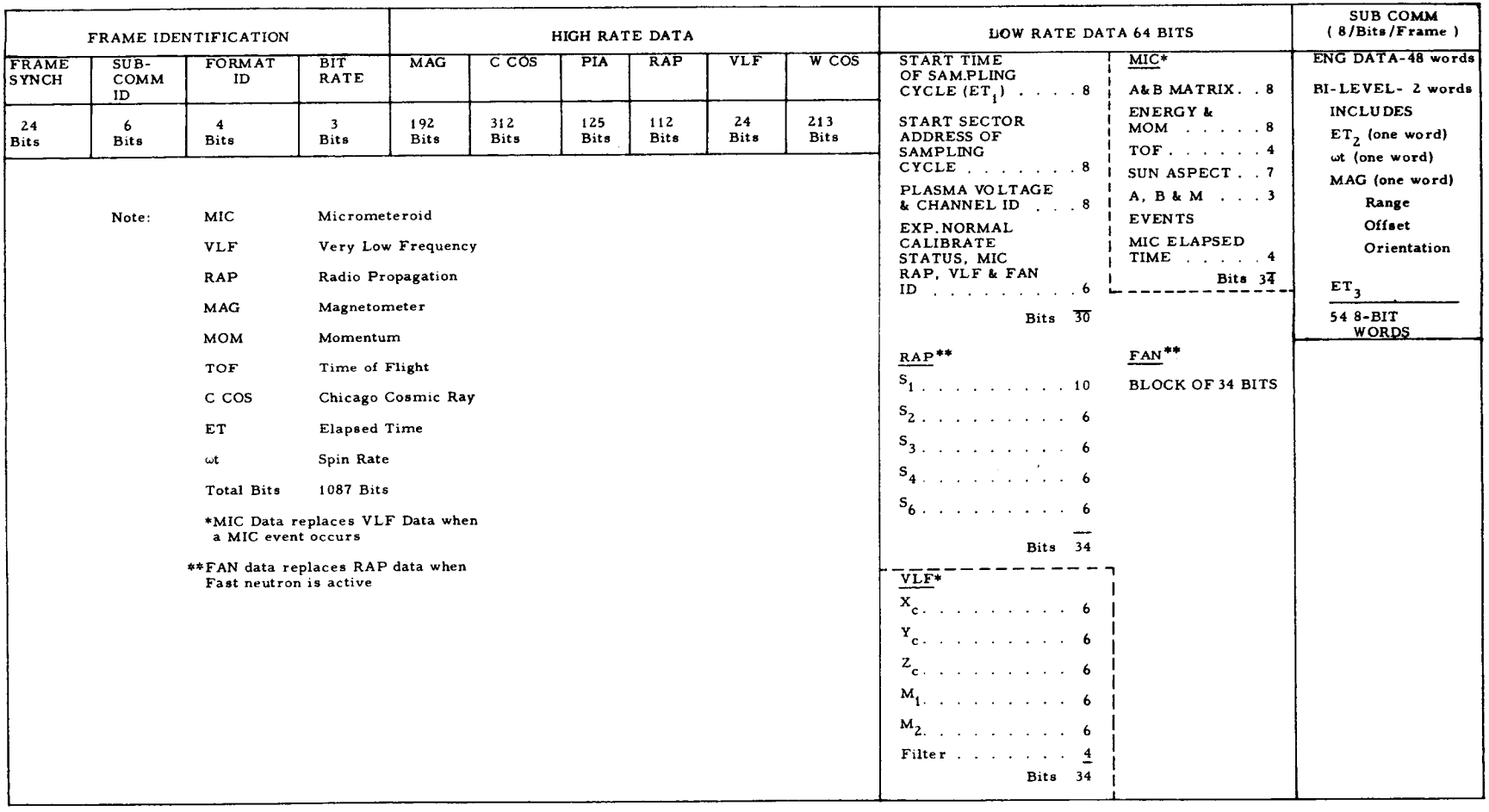


Figure 3-10. Typical Format (1087-Bit Format Size)

intervals. Eight samples are taken per spacecraft revolution in this sample. Sample and hold gates are used to sample all three axes at the same instant in time in this case.

Chicago Cosmic Ray. Four samples are taken per revolution, each sample containing 74 bits. In addition 16 bits of sector data are transmitted since different sectors are sampled each revolution. That is, a group of successive sectors are sampled each revolution until all sectors are sampled. A total of 312 bits are allocated for the cosmic ray experiment.

Plasma Probe. Plasma flux is measured 16 times per revolution and is encoded to 7-bit accuracy requiring a total of 112 bits. In addition, a maximum flux mode reading is also provided which requires 13 additional bits for a total scan bit requirement of 125 bits for this format.

Radio Propagation. S5 (49.8 MHz amplitude) is sampled 16 times per spacecraft revolution to an accuracy of 7 bits, thus totaling 112 bits.

Very Low Frequency. The E_x and E_y electric field components are sampled and encoded to 6-bit accuracy two times per revolution. Thus 24 bits are required.

Webber's Cosmic Ray. This cosmic ray experiment is sampled twice per revolution, accumulating 213 bits total.

LOW RATE DATA Low rate data consists of experiment data that is sampled only once per revolution or less. The following data is transmitted once per revolution.

Start Time of Sampling Cycle (ET_1). This time refers to the recording of elapsed time at the beginning of each spacecraft sampling cycle. This elapsed time (8 bits) has a resolution of 1 second.

Start Sector Address of Sampling Cycle. These 8 bits specify the address of the first sector which is sampled each spacecraft revolution. This address is required to determine the relative time in which the samples were taken. See

Section 3.3.3 for discussion. For example, if sampling begins with sector 5 and ends with sector 4 for one complete spacecraft revolution, it may be necessary to know that sector 4 was sampled 1 second after sector 5.

Plasma Channel and Voltage ID. These 8 bits specify which of the eight channels and 24 analyzer plate voltages were used to make the plasma flux measurement.

Experiment Normal/Calibrate Status, MIC, RAP, FAN, and VLF ID. Four bits of this group are allocated to indicate experiment normal/calibrate status, and 2 bits identify which of the four groups of 34 bits are being transmitted (MIC, RAP, FAN, or VLF).

RAP, VLF, FAN, and MIC Experiment Data. These four groups each contain 34 bits; only one of the four groups is transmitted in one frame. RAP and VLF data are transmitted alternately in each frame. MIC data is transmitted in place of VLF data when a MIC event occurs. This is expected to be in the order of ten events per day. Fast neutron data will be transmitted alternately with RAP data when fast neutron data is available.

Subcommutated Data. Subcommutated data is transmitted one 8-bit word each revolution. There are 64 words in the subcommutator; 48 engineering data, including two bi-level words. In addition to engineering data elapsed time (ET_2), the second most significant 8-bit groups are: spacecraft spin rate (ωt); magnetometer orientation, offset and orientation, and elapsed time (ET_3). The least significant 8-bit group is transmitted within the subcomm elapsed time (ET_3). This is transmitted in the place of magnetometer data once each 24 hours. One of the 8 bits allocated for (ωt) identifies which group (ET_3 or mag) is transmitted.

The resultant frame size for the example format (Figure 3-10) is 1088 bits. The frame sizes are programmable and any practical frame size may be used for each of the eight transmission bit rates. Typically, frame sizes in the vicinity of 512 to 1200 bits will be used for the CDS

(see Appendix B). The only criterion is that, whatever frame size is chosen for a given transmission bit rate, a complete frame of data must be processed, formatted, and stored within the output buffer in time for continuous uninterrupted transmission.

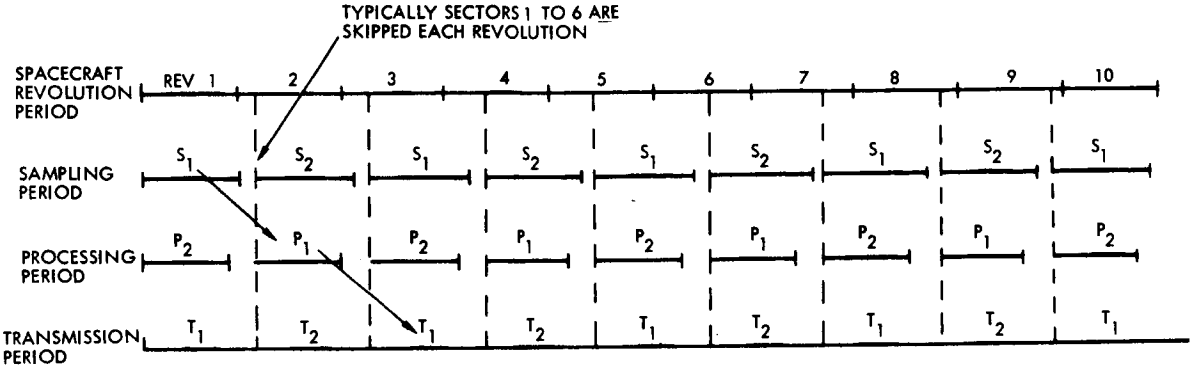
3.3.3 Sun Pulse and Data Correlation

The feature of a programmable size format, however, necessitates that special attention be given to spacecraft timing. In order to synchronize data sampling with data transmission, data sampling is initiated by a marker signal from one of the output buffers. A marker signal is generated when the contents of an output buffer is completely read out. Initiating the sampling for each spacecraft with a marker signal from the output buffers insures that data will be sampled, processed, and stored within the output buffers in time for uninterrupted transmission.

The timing relationship between sampling and transmitting of data is given in Figure 3-11. The timing line labeled "spacecraft revolutions period" is marked at increments indicating the period of one revolution. The timing line labeled "processing period" illustrates the time required to process, format, and store data into the output buffer for transmission. Note that the processing period is shorter than the sampling and transmission period. This is to insure that data is processed and stored within the output buffer in time for uninterrupted transmission. The timing line labeled "transmission period" represents the time required to transmit data from one output buffer. Note that this timing line is not broken, indicating uninterrupted data transmission.

The transmission period is determined by the transmission bit rate and the number of bits stored within an output buffer. As indicated in the figure, the transmission period is made longer than the sampling period (one spacecraft revolution). Sampling and processing are initiated when transmission from either of the two output buffers is complete. This insures that data is sampled from one complete revolution each sampling period and that data is processed, formatted, and stored within the output buffers in time for continuous uninterrupted transmission.

Figure 3-11 indicates a gap between sampling periods which is a result of this type of timing scheme. The length of the gap is dependent



S_1 - SAMPLE AND STORE DATA INTO IDB 1

S_2 - SAMPLE AND STORE DATA INTO IDB 2

P_1 - PROCESS STORED DATA FROM IDB 1

P_2 - PROCESS STORED DATA FROM IDB 2

T_1 - TRANSMIT 1

T_2 - TRANSMIT 2

Figure 3-11. CDS Timing Diagram

upon the size of the programmable frame or format and is typically one to six sector periods. Skipping six sectors each revolution, for example, insures proper timing for spin rate changes in the order of 2 rpm. If the spin rate should exceed 2 rpm, the length of the format (transmission period) is increased to again provide proper timing.

The skipping of one to six sectors per sampling cycle is felt to be of little consequence since:

1) The number of sectors skipped is known.

2) Data is never transmitted from each of the 64 sectors so that the granularity of data remains virtually unchanged. For example, when data is sampled four times per revolution, the skipping of two sectors means that, in realtime, one of the four samples will be separated by 18 sectors instead of 16 sectors. The advantages of this timing scheme are

   - Data is transmitted by fixed selected bit rates, thus simplifying DSIF processing
   - Frame size is programmable, maximizing data transmission efficiency (no fill-in bits required within the format)
   - Minor changes in spin rate (1-2 rpm) are compensated for automatically.

3) Major changes in spin rate (50 percent) can be compensated for by re-programming (changing the size of the formats).

The processing period (Figure 3-11) includes the time required to shift bulk store data into the output buffer. That is, when bulk store data is interleaved with realtime data for transmission, the output buffer is filled with one frame of realtime data and one frame of bulk store data each second.

3.3.4 Special CDS Operations

3.3.4.1 Self Test

A self test routine is provided within the processor instruction memory which may be called upon automatically, perhaps at daily intervals, to evaluate the performance of the CDS. A special routine will check offset and calibration of the experiments; another routine will test A/D converter accuracy by encoding three known voltages and comparing them against stored limits. The data operator will be checked for proper addition, subtraction, shifting, etc. The results from each test may be transmitted to the ground or a simple test OK code may be transmitted for the entire self test routine within the subcomm.

3.3.4.2 Parity Check

The processor memory is checked for parity each time an instruction is read. If a parity error occurs, the CDS automatically enters the fixed realtime transmission mode which will completely bypass the processor. The ground station is notified that a parity error has occurred via realtime transmission. The ground station may then command switching to a redundant processor instruction memory or a program verification mode may be commanded where the contents of the program in use are transmitted to the ground. If only a few commands are in error, they may be re-entered into the processor instruction memory via uplink transmission.

3.3.4.3 Bulk Store Operations

A bulk store magnetic tape unit, with a capacity of approximately 10^7 bits is provided to store data during periods of noncommunication with DSIF stations. When communications are re-established, realtime and stored data may be interleaved for transmission.

3.3.4.4 Transmission Bit Rates

The CDS can transmit data at eight bit rates: 2048, 1024, 512, 256, 128, 64, 32, and 16 bps. At the present time three programs will be used to process data at the different transmission bit rates. For example, one program may perform processing at 2048 and 1024 bps, a second program at 512, 256, and 128 bps, and a third program at

64, 32, and 16 bps (refer to Appendix B). The programs are not exclusively for one set of transmission bit rates. For example, the processor program normally allocated for the 1024 bps transmission rate can be used at the lower bit rate of 64 bps. However, with this combination 16 spacecraft revolutions may be required to transmit one frame of data.

3.3.4.5 Convolutional Coding

The CDS output will be convolutional coded using a coding technique suggested by NASA/ARC. This coding technique adds one code bit for each data bit. The code is generated on the basis of 24-bit data blocks.

3.3.5 Elapsed Time

The CDS is designed with a central elapsed time clock. The clock consists of a flip-flop countdown chain which divides down the output from the master oscillator. The elapsed time clock has a resolution of 1 second with a worst case long term stability of $\pm$ 0.01 percent for a 2-year period. The elapsed time counter consists of 24 flip-flops providing a recycling time of approximately 194 days.

Elapsed time is transmitted with each frame. The eight least significant bits (ETI), 64 seconds recycling time, is transmitted within the low rate data group. The next most significant group (ET_2), approximately 18 hours recycling time, is transmitted in the subcomm; ET_2 is therefore transmitted once each 64 spacecraft revolutions or once each 64 seconds, assuming a 1-rps spin rate. The most significant 8-bit group (ET_3), 194 day recycling time, is transmitted once every 24 hours. This distribution of elapsed time transmission provides adequate data correlation with elapsed time under all modes of operation.

4. CENTRAL DATA SYSTEM DETAILED DESIGN DISCUSSION

4.1 INPUT DATA SAMPLER

The basic requirement of the input data sampler is to sample CDS input signals, count pulses, encode signals where required, and store this data in preassigned input data buffer (IDB) locations for processing by the processor.

The following sections describe in detail the mechanization of the input data sampler (IDS).

4.1.1 Clock Generator

The clock generator provides the basic timing pulses to the data sampler and to the processor. As shown in Figure 4-1, the clock generator consists of a:

- Master oscillator
- Sun sector generator
- Bit rate generator
- Elapsed time counter
- Sampler control logic.

The master oscillator operating at 131.072 kHz is crystal controlled. This output is fed directly to the sun sector generator, the bit rate generator, and the processor. The clock signals to the data sampler and to the elapsed time counter are derived from the bit rate generator. Figure 4-2 shows a simplified diagram of the mechanization of the master oscillator.

The sun sector generator divides each spacecraft revolution into 64 sectors. Each sector is identified (coded) based on the position of the spacecraft sun sensor with respect to the sun. Thus, code "000000" identifies the first of the 64 sectors (Figure 4-3). These sector codes identify the position of the spacecraft when a particular input is sampled.

The sun sector generator (Figure 4-4) consists basically of counters and associated gates. The 131 kHz clock from the master oscillator is first divided down by the desired number of increments.

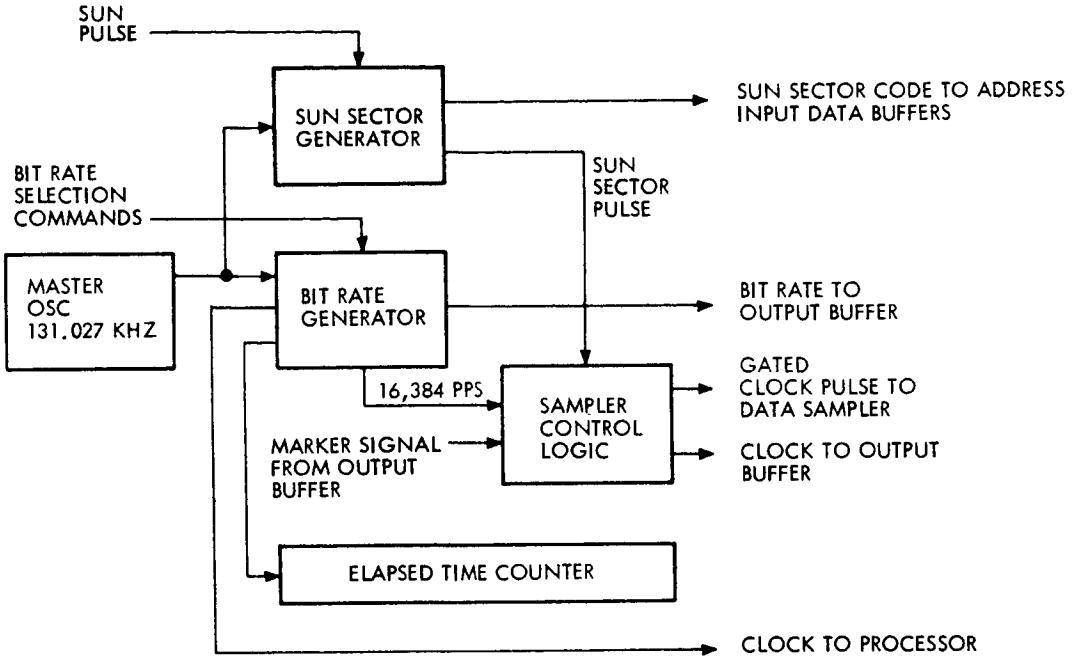


Figure 4-1. Clock Generator Block Diagram

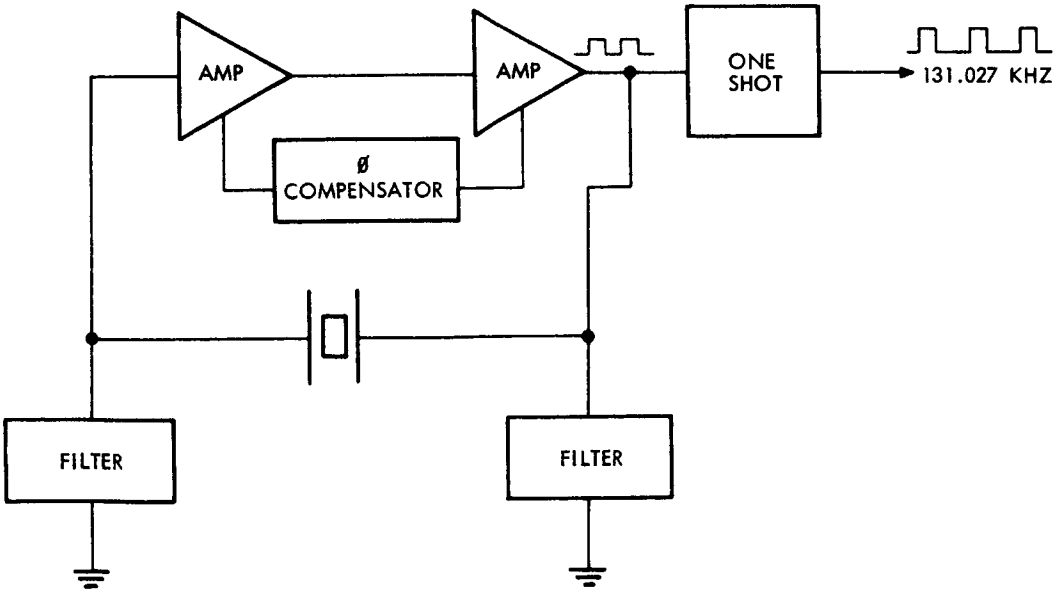


Figure 4-2. Master Oscillator

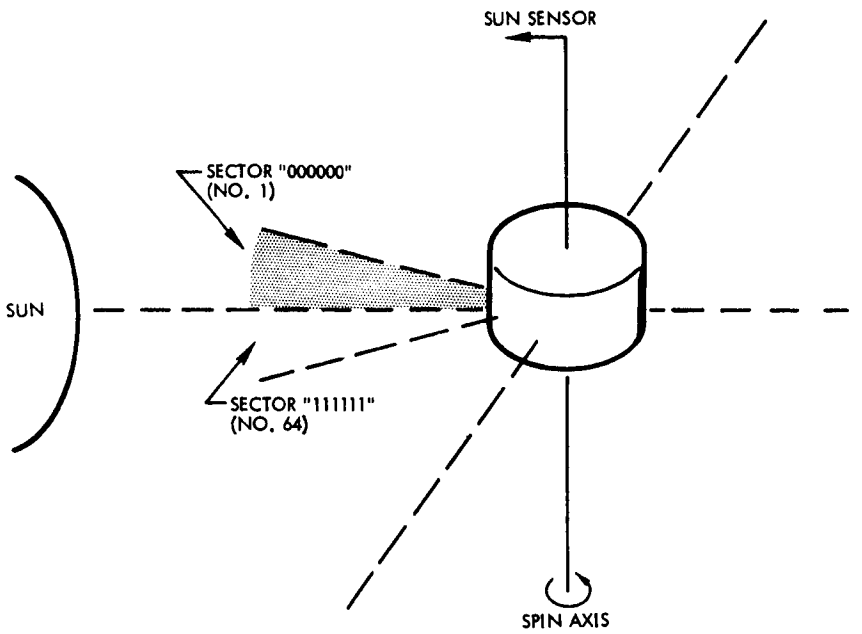


Figure 4-3. Sector Identification

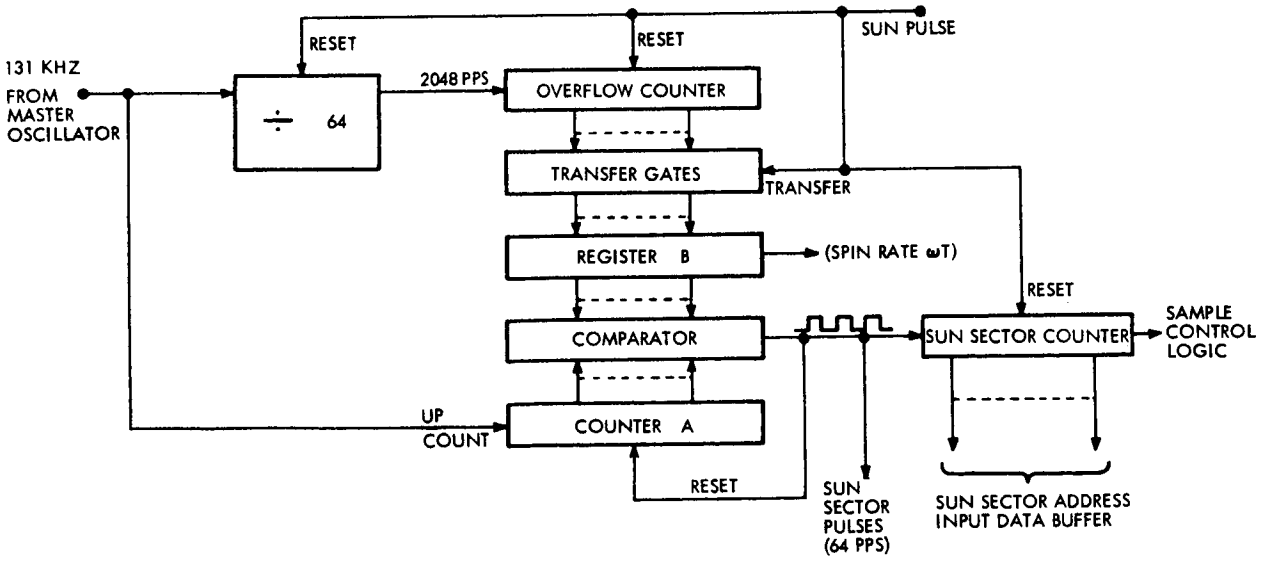


Figure 4-4. Sun Sector Generator Simplified Block Diagram

That is, if n sectors are desired per spacecraft revolution then a modulo n is used. In this case, a divide-by-64 is shown. This is followed by an overflow counter. Assuming a spacecraft revolution of 1 rps, an 11-bit counter is required for this part to store 2048 bits.

$$\frac{131 \text{ kbps}}{64} = 2048 \text{ bps}$$

However, to cover the condition where the spacecraft is spinning less than 1 rps, a 12-bit counter is provided since more than 2048 counts will be generated. At every sun pulse, the overflow counter is reset and its content transferred to register B. Register B thus contains a number which represents the spin rate (ωt) of the spacecraft. The accuracy of spin rate determination is 0.05 percent. This is based upon a time base of 2048 bps.

Meanwhile, the 131 kbps signal is fed to another counter (counter A). The content of counter A is compared to that of register B. Whenever the count in the B register coincides with the count in the A register, a pulse is generated which is the sun sector pulse. This sun sector pulse is also used in resetting counter A. Thus, for a spacecraft spinning at 1 rps, 64 sun sector pulses per second are generated, thus obtaining 64 increments in a spacecraft revolution.

The bit rate generator consists of a series of dividers and associated gates as shown in Figure 4-5. By ground commands, any one of the following eight bit rates can be selected:

- 2048 bps
- 1024 bps
- 512 bps
- 256 bps
- 128 bps
- 64 bps
- 32 bps
- 16 bps

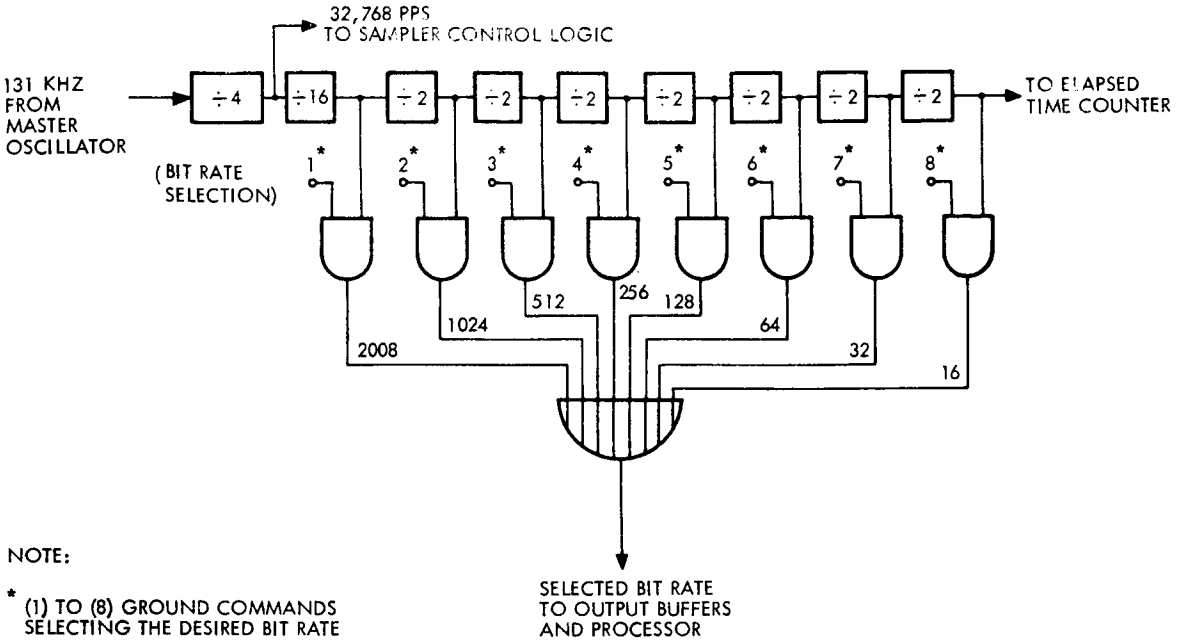


Figure 4-5. Bit Rate Generator

Figure 4-6 indicates the logic which controls the clock pulses to the fixed programmer. This logic insures that sampling is initiated at the beginning of a sector and also insures that sampling is initiated following the end of each output buffer transmission cycle. As previously explained, sampling is initiated whenever the output buffer is completely read out. This is shown in Figure 4-6 by the signals M1 and M2 (marker signals) derived from the output buffers. The timing diagram of Figure 4-7 shows how the clock pulse is gated to the fixed programmer. Either M1 or M2 will set flip-flop S2. This then enables the set gate of flip-flop S3, and upon receipt of the next sun sector pulse, S3 goes TRUE. The clock pulse is then gated through to the fixed programmer until S2 and S3 are reset by the stop signal.

4.1.2 Fixed Programmer

The fixed programmer (Figure 4-8) consists of:

- The timing generator
- The main frame word counter
- The subcomm or subframe word counter.

The timing generator develops T1 through T8 timing pulses used for A/D converter operation. In a typical application, at T1 time, an analog gate is enabled. As shown in Figure 4-9, the analog is allowed to stabilize within the first bit time prior to encoding. A/D conversion will be performed during bit times T1 through T8 to an 8-bit encoding accuracy. Then at T8 time, the encoded data is entered into the IDB. The sampling rate is determined by decoding the main frame and subframe counters (Figure 4-8). The main frame word counter contains 6 flip-flops to accommodate selecting up to 64 inputs requiring high rate sampling (64 times per spacecraft revolution). Presently, only 36 words appear adequate to cover those data listed in Research Report No. 2. The fast neutron data is not included since it is sampled and handled in a special manner as discussed later within this section.

The subframe or subcomm word counter is used to select those inputs requiring very low sampling (e.g., engineering type data).

As shown in Figure 4-8, the main frame word counter and timing generator is reset by the sun sector pulse. This is to insure that sampling is initiated at the beginning of each of the 64 sectors.

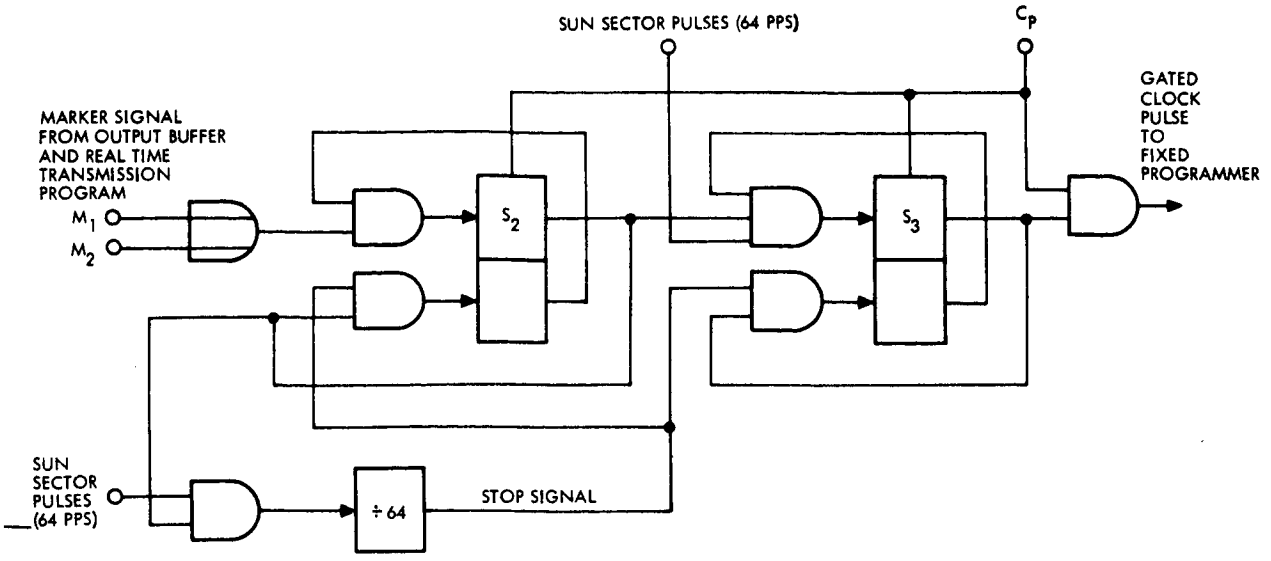


Figure 4-6. Sampler Control Logic

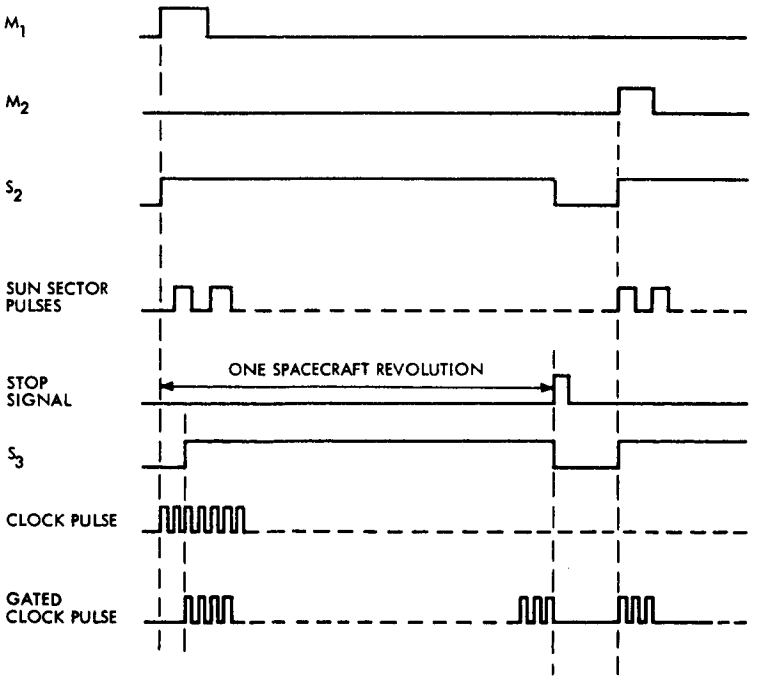


Figure 4-7. Timing Diagram of Control Logic

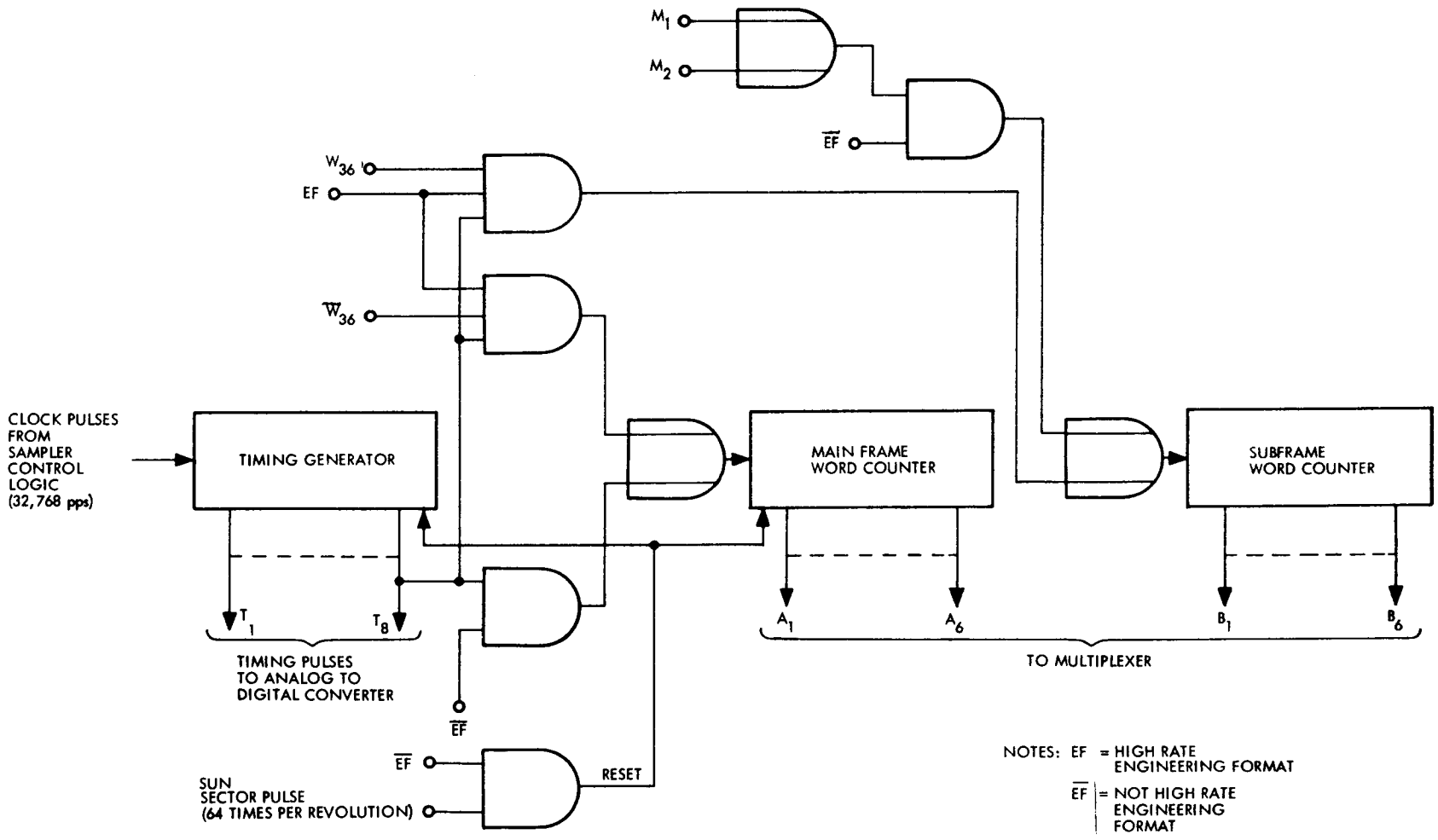


Figure 4-8. Fixed Programmer, Simplified Diagram

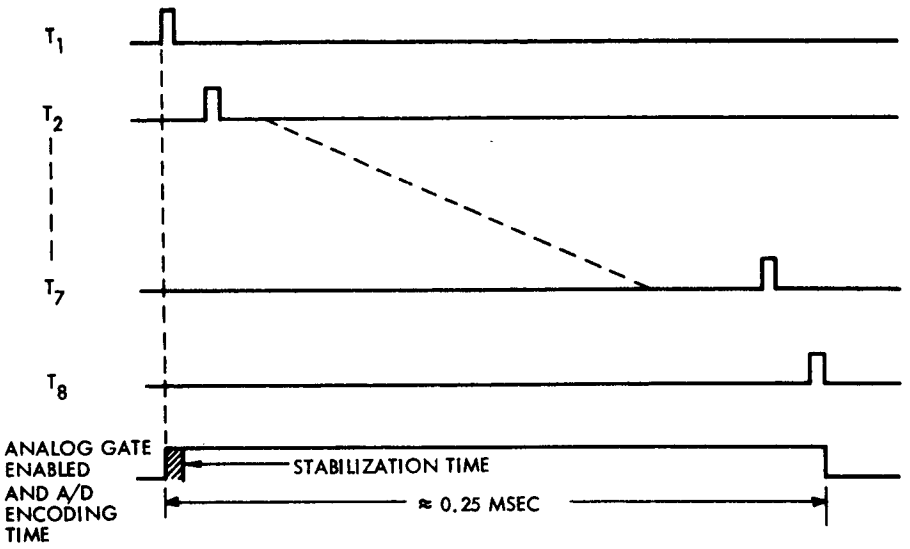


Figure 4-9. Typical Timing Diagram Related to the T Pulses

During all programmable modes of operation the main frame word counter is sequentially stepped at every T8 pulse (the word rate pulse). The word rate pulse is gated through by the controlling signal designated $\overline{EF}$ (not high rate engineering format). However, during the fixed realtime engineering format mode of operation, the main frame word counter counts up to 36 (word 36 is assigned for engineering data) and remains at this position until commanded to return to the programmable mode. During the time the main frame word counter is held at word 36, the word rate signal (T8) is gated directly to the subframe word counter. This then permits the subframe word counter to operate at a main frame rate. But specifically this technique allows sampling of only engineering type data.

The input to the timing generator is the 32, 768 pps clock pulse generated by the bit rate generator. This rate was derived by determining the number of inputs which must be sampled each sun sector. Taking 64 as the possible number to be sampled each sector and with each word requiring eight T pulses, the clock pulse rate is calculated to be:

$$Cp = \frac{64 \times 8}{\text{duration of } 1/64 \text{ of a revolution}} = 32,768$$

A higher clock rate would be required to accommodate a greater number of inputs or a higher sampling rate.

4.1.3 Multiplexer and Gates

The main and subframe multiplexer (Figure 4-10) decodes the states of the fixed programmer register to provide control signals to the input data gates. The main frame multiplexer operates at the highest desired sampling rate and controls those gates associated with the experiment data. The subframe multiplexer, on the other hand, operates at a very low rate and controls those gates associated with engineering type data. Figure 4-11 shows the 64 main decoding gates with the

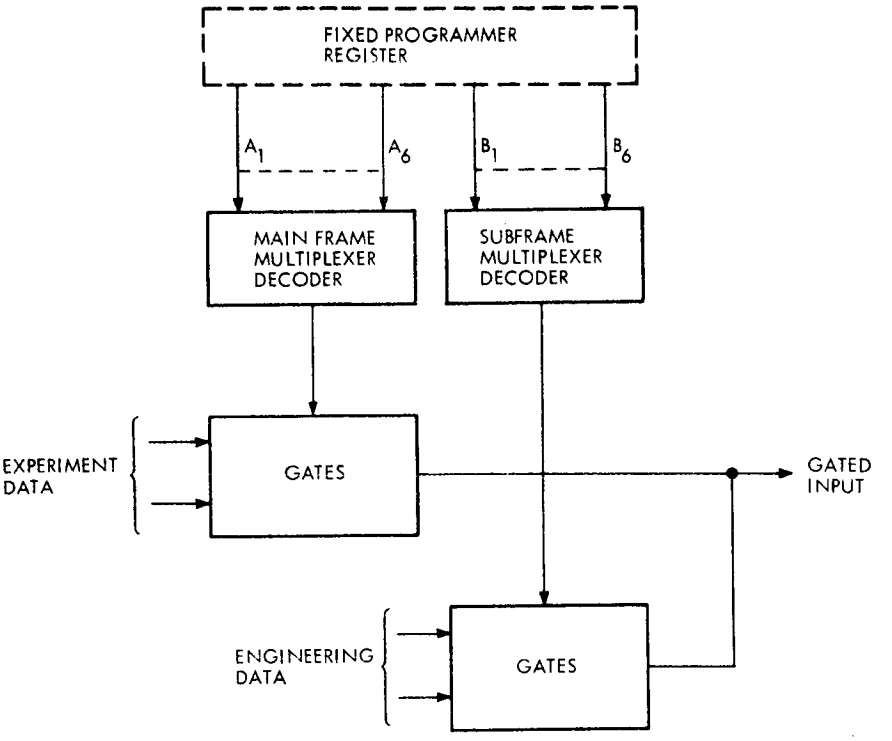


Figure 4-10. Main and Subframe Multiplier Block Diagram

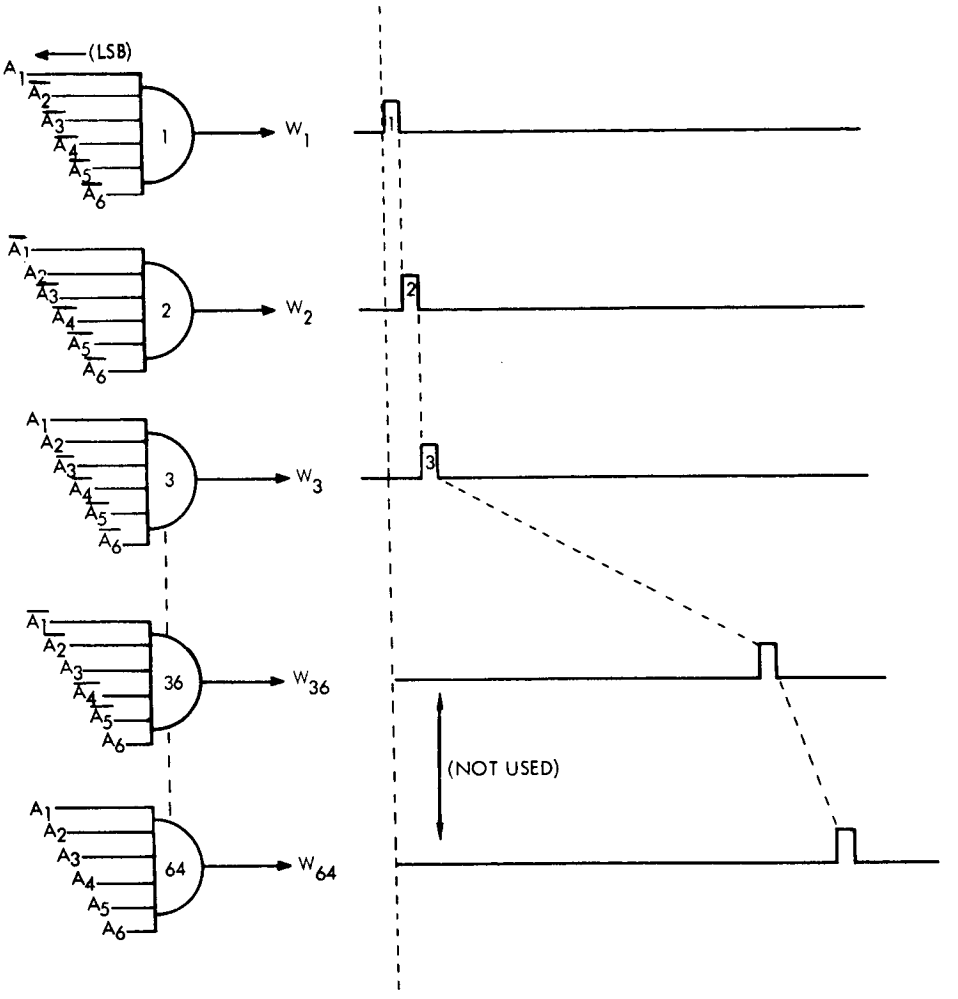


Figure 4-11. Main Multiplexer Decoding Gate with Timing Diagram

associated timing diagram to illustrate their sequence of events. The organization of the submultiplexer is similar to the main multiplexer except for a larger number of required gates. Figure 4-12 illustrates the analog gating and decoding for subcommutating techniques. As shown, the analog inputs are divided into four groups of 16 analog gates followed by one group of four analog gates (2nd level gating). The advantages of this gating technique are to minimize error due to leakage and to minimize failure modes. Figure 4-13 shows a typical gate for each type of input data signal (i.e., analog, bi-level, and pulse data). Figures 4-14 through 4-18 illustrate the mechanization of data sampling. Figure 4-14 shows the gating arrangement of the high rate analog data which is sampled every sector.

Figure 4-15 illustrates the scheme for sampling Webber's cosmic pulse data, while Figure 4-16 illustrates pulse gating for Chicago cosmic pulse data. As shown, the gates are enabled (counts accumulated) through each sector except for the particular word time that is used to store the contents of the counters within the IDB. For example, Chicago cosmic pulse data signal $D_1 \ \overline{D}_2 \ \overline{D}_4$ is accumulated over a period of one sector and then stored into the IDB during word 31.

Figures 4-17 and 4-18 illustrate the gating for the sampling of low rate data. This data is sampled once per revolution during word time 35 within the sectors indicated in the figures. Figure 4-12 illustrates the sampling of engineering data. During programmed realtime science operation, engineering data is sampled once per revolution. Consequently, with 64 inputs, complete scanning will require 64 spacecraft revolutions. However, when the CDS is commanded to operate in the high rate engineering format, the multiplexer will remain in word 36 and the submultiplexer will operate at a stepped up rate, sampling only engineering data. This engineering data includes experiment temperature data, spacecraft performance data, elapsed time, and spin rate.

Figure 4-19 is a basic block diagram for the required experiment control signals. As shown, the plasma voltage steps are changed by the sun pulse.

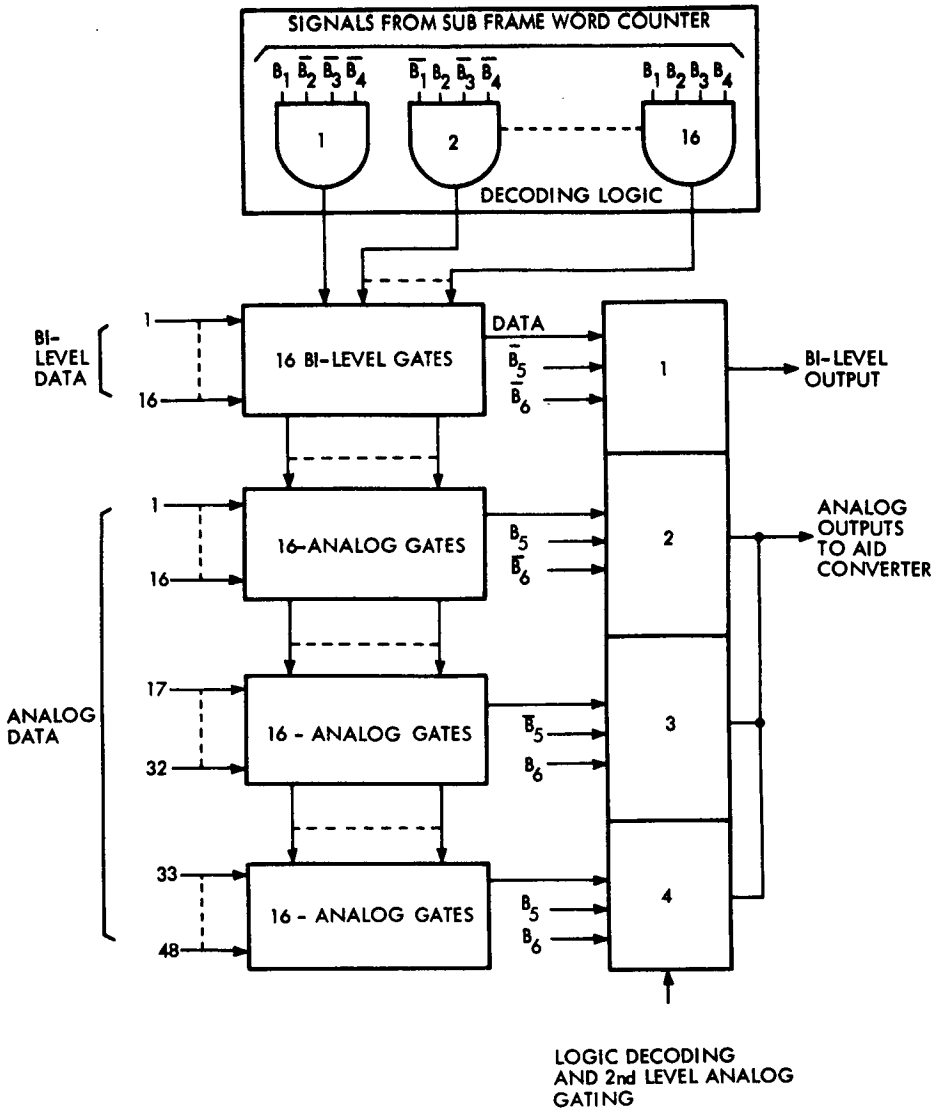


Figure 4-12. Submultiplexer with Second Level Gating

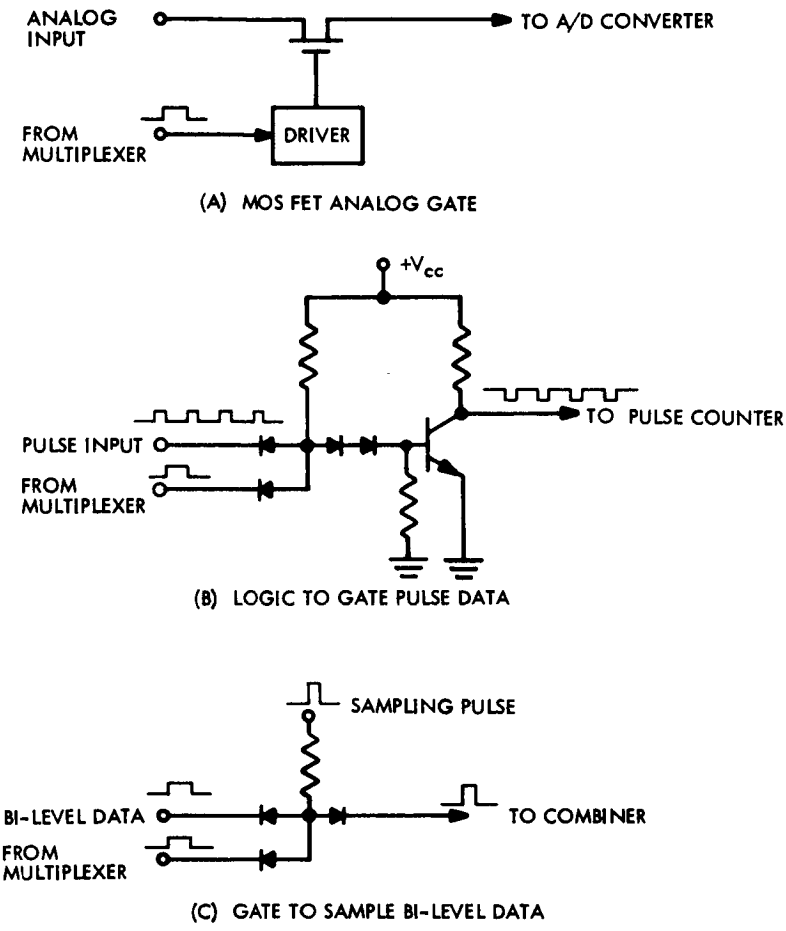


Figure 4-13. Typical Input Gates

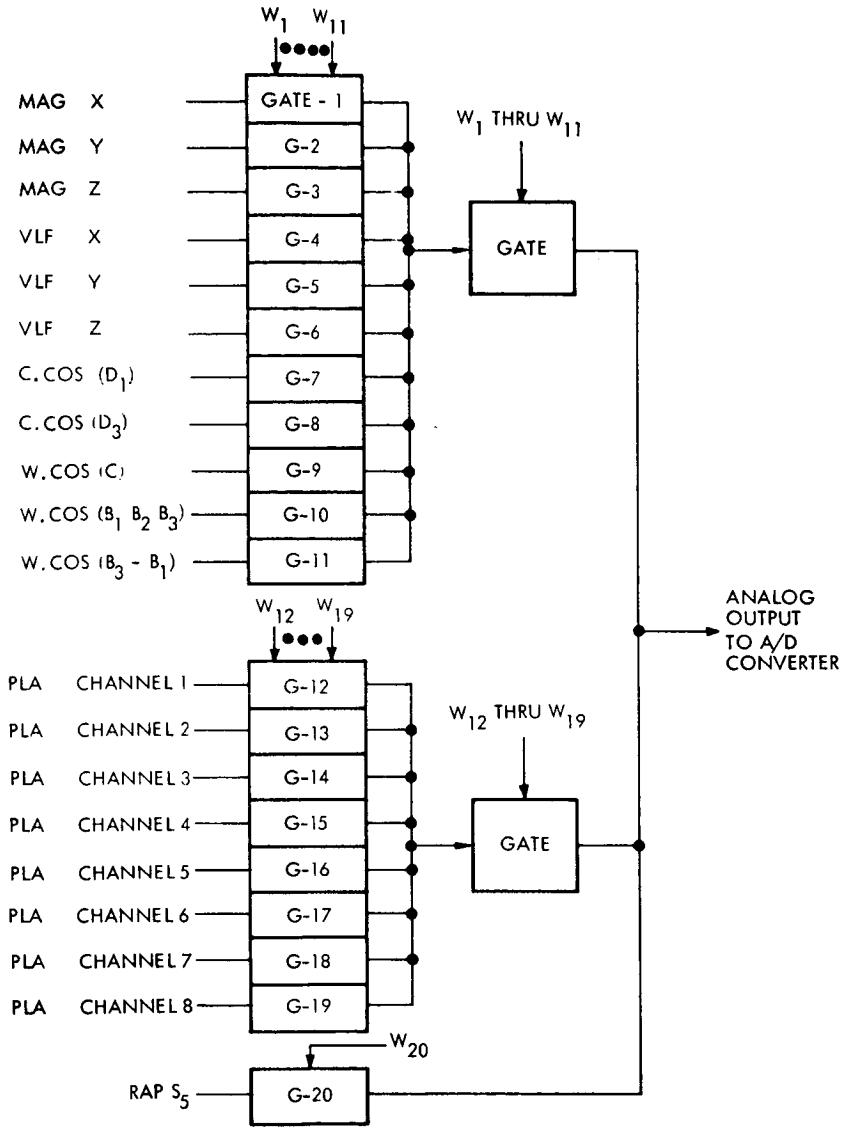


Figure 4-14. Organization of Analog Gates, Main Multiplexer

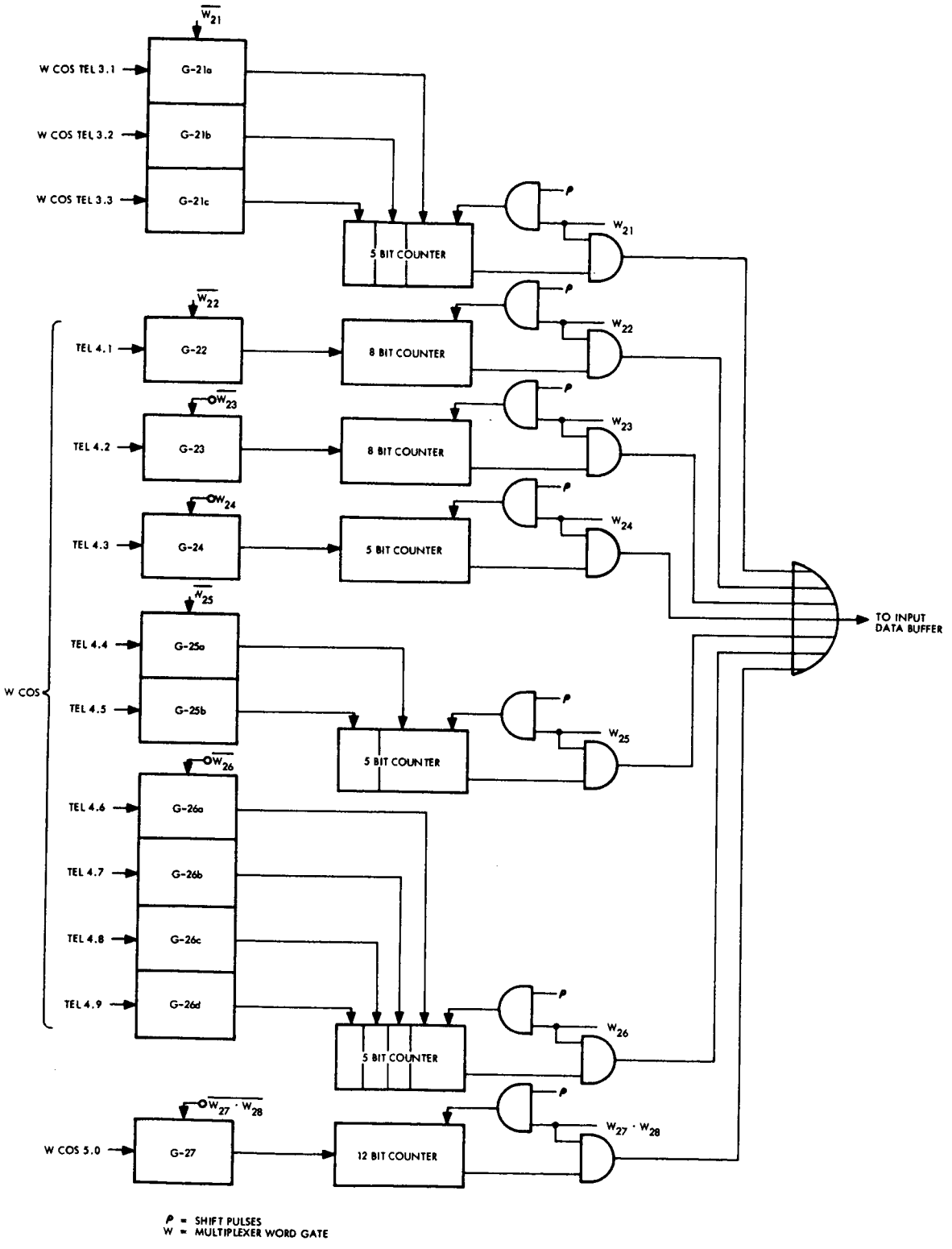


Figure 4-15. Pulse Data Sampler for Webber's Cosmic Ray Experiment

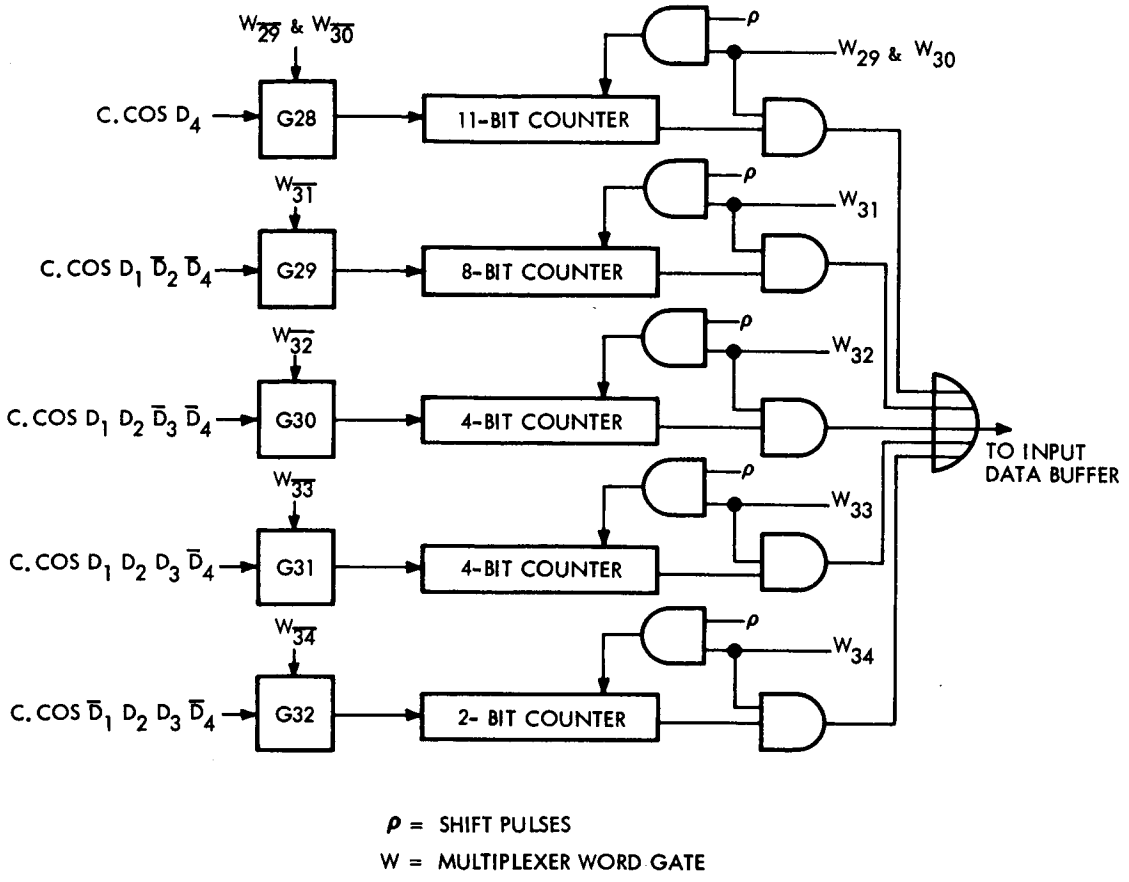


Figure 4-16. Pulse Data Sampler for Chicago Cosmic Ray Experiment

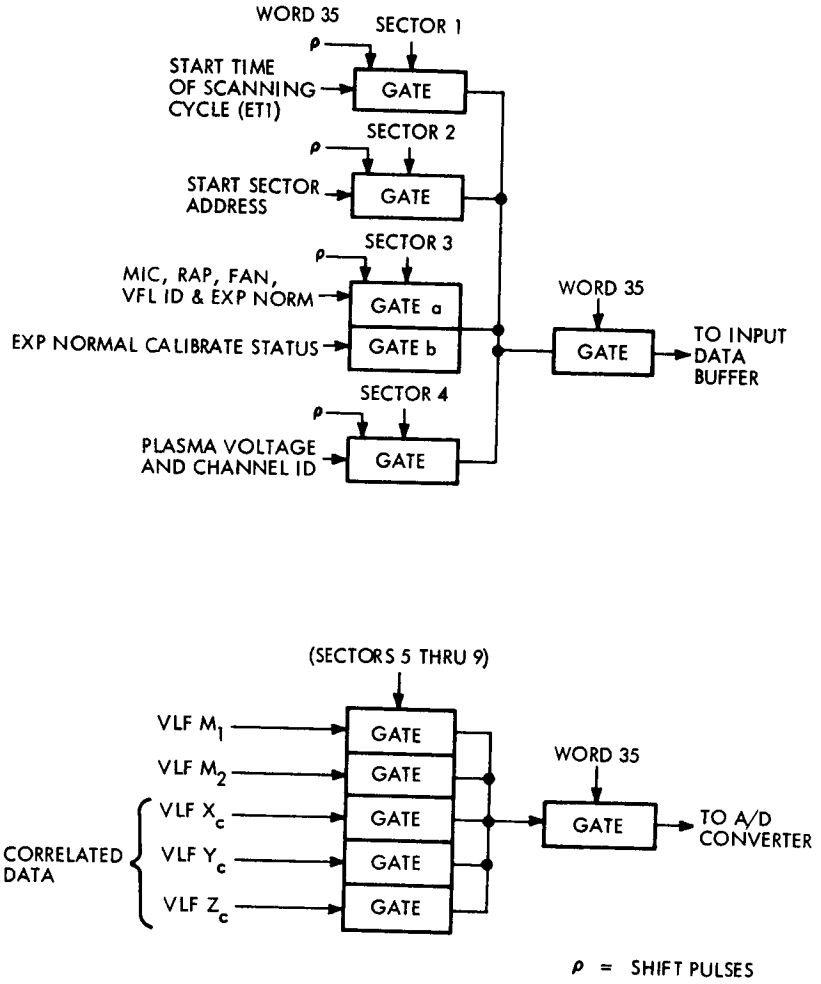
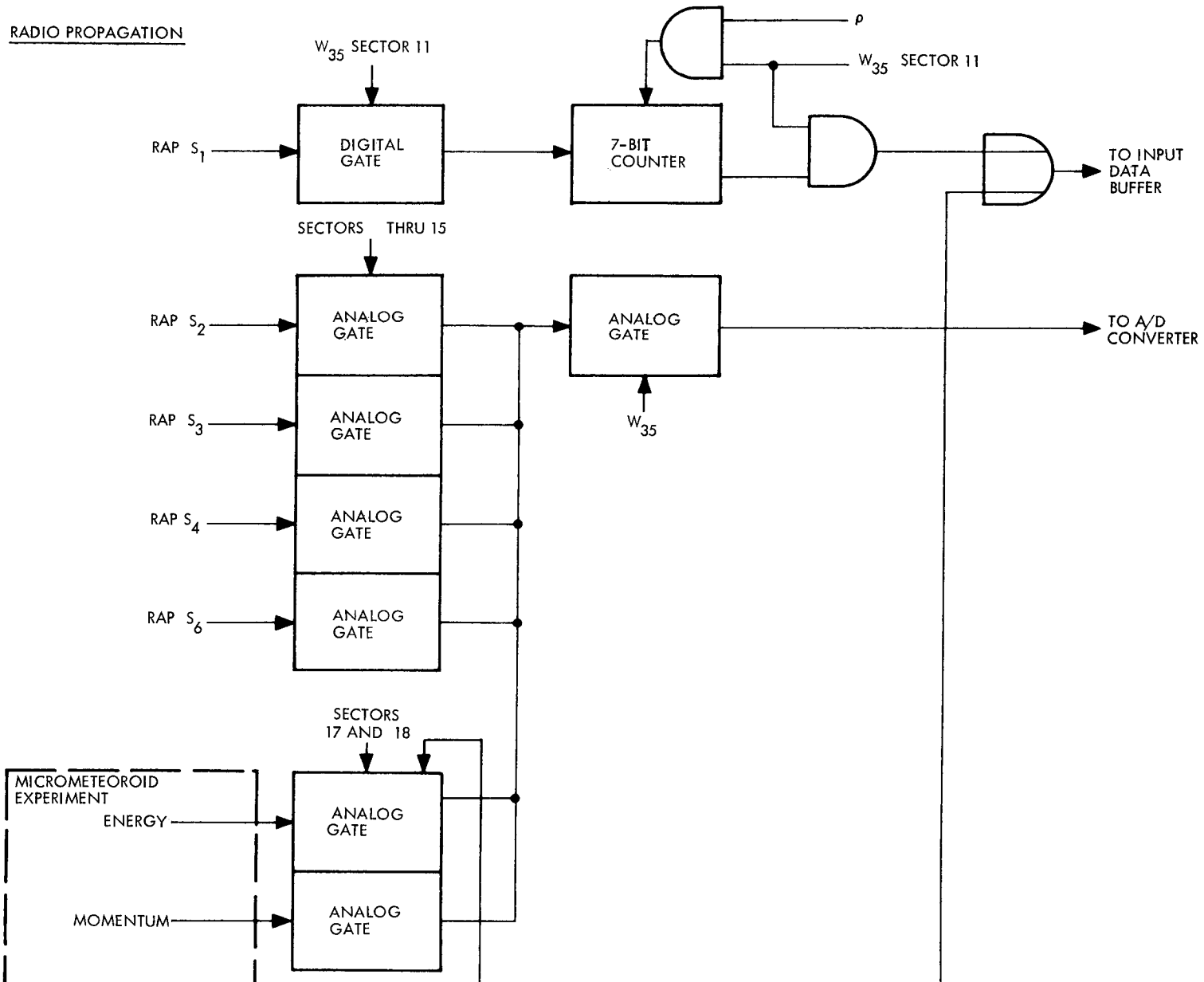


Figure 4-17. Sampling of Low Rate Data


RADIO PROPAGATION
W_{35} SECTOR 11
RAP S_1
DIGITAL GATE
7-BIT COUNTER
ρ
W_{35} SECTOR 11
TO INPUT DATA BUFFER
SECTORS THRU 15
RAP S_2
ANALOG GATE
ANALOG GATE
TO A/D CONVERTER
RAP S_3
ANALOG GATE
W_{35}
RAP S_4
ANALOG GATE
RAP S_6
ANALOG GATE
SECTORS 17 AND 18
MICROMETEOROID EXPERIMENT
ENERGY
ANALOG GATE
MOMENTUM
ANALOG GATE

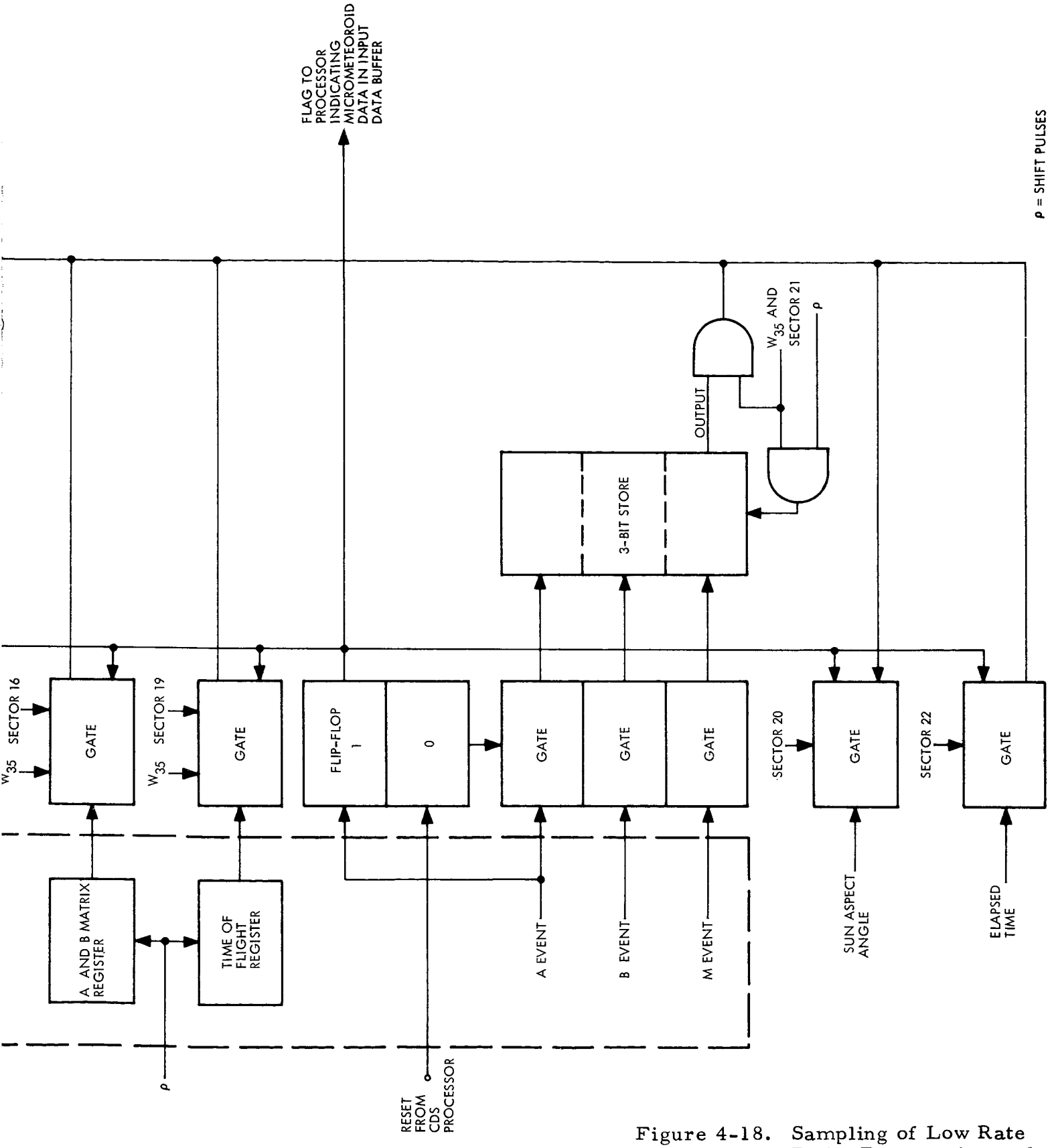


Figure 4-18. Sampling of Low Rate Radio Propagation and Micrometeoroid Data

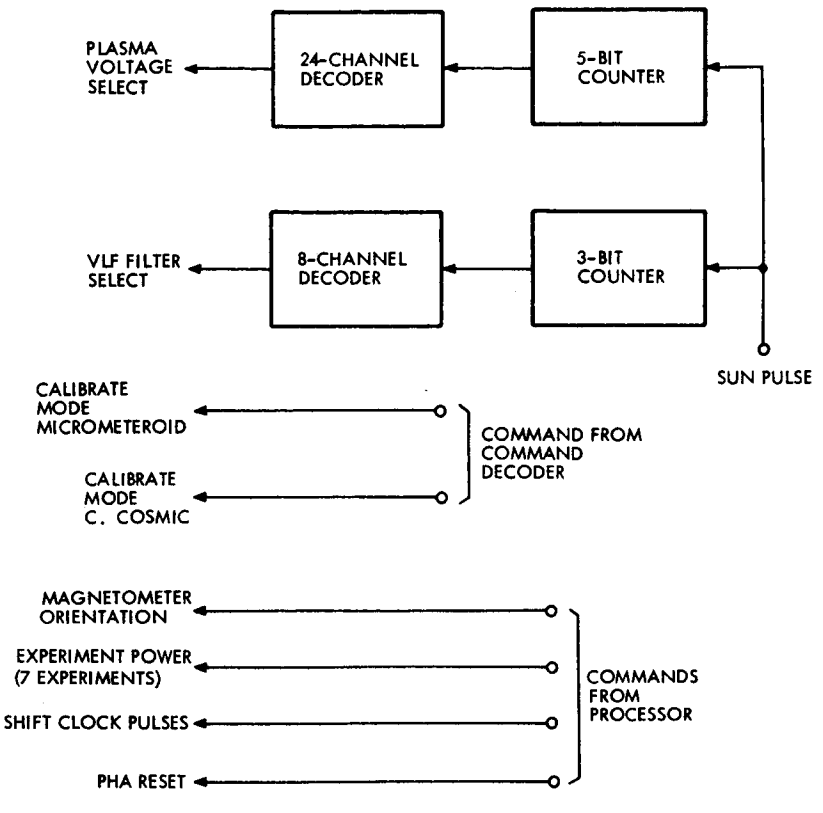


Figure 4-19. Experiment Control Block Diagram

4.1.4 Input Data Buffer

Two IDB's are used. This permits data to be sampled during each spacecraft revolution. That is, one IDB fills while data is being extracted from the other IDB for processing. Both IDB's are used only during the realtime scientific only mode. Figure 4-20 illustrates the logic for this operation. The use of two IDB's also provides for a pseudo redundant mode of operation to increase system reliability. That is, a special mode may be commanded which will bypass one of the two IDB's. Data will then be entered into the one selected IDB during one spacecraft revolution and read out for processing during the next.

Figure 4-21 illustrates the logic required for addressing the IDB. As shown, the IDB is addressed by the fixed programmed counter, the processor, the spacecraft sector generator, and by the fixed realtime programmer.

4.2 PROCESSOR

4.2.1 General

The processor (Figure 4-22) consists of:

- Processor instruction memory
- Data operator and processor logic
- Auxiliary memory.

All processor operations are under the control of a series of programs stored within the processor instruction memory. During normal modes of operation the processor selects data from the IDB, processes and formats the data by use of the data operator, and stores the processed data into the output buffer.

Since the instruction memory controls all processor operations, high reliability must be obtained within the memory and associated circuitry to insure mission success. A high confidence level is insured for the instruction memory by the use of a redundant instruction memory as shown in Figure 4-22.

The instruction memory stores the programs for the five programmable modes of operation. Seven basic programs are required for these five modes as shown in Figure 4-23. This figure illustrates the

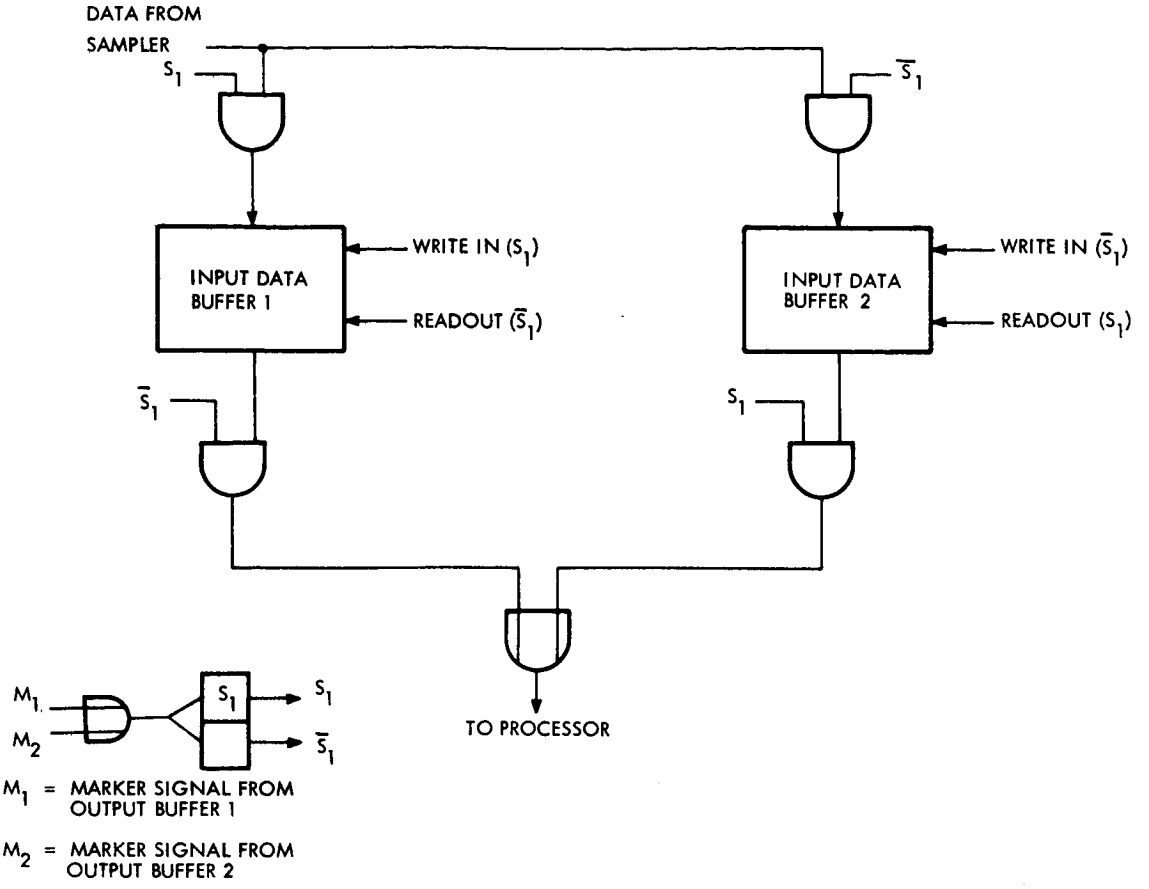


Figure 4-20. Write in and Readout of the Two Input Data Buffers

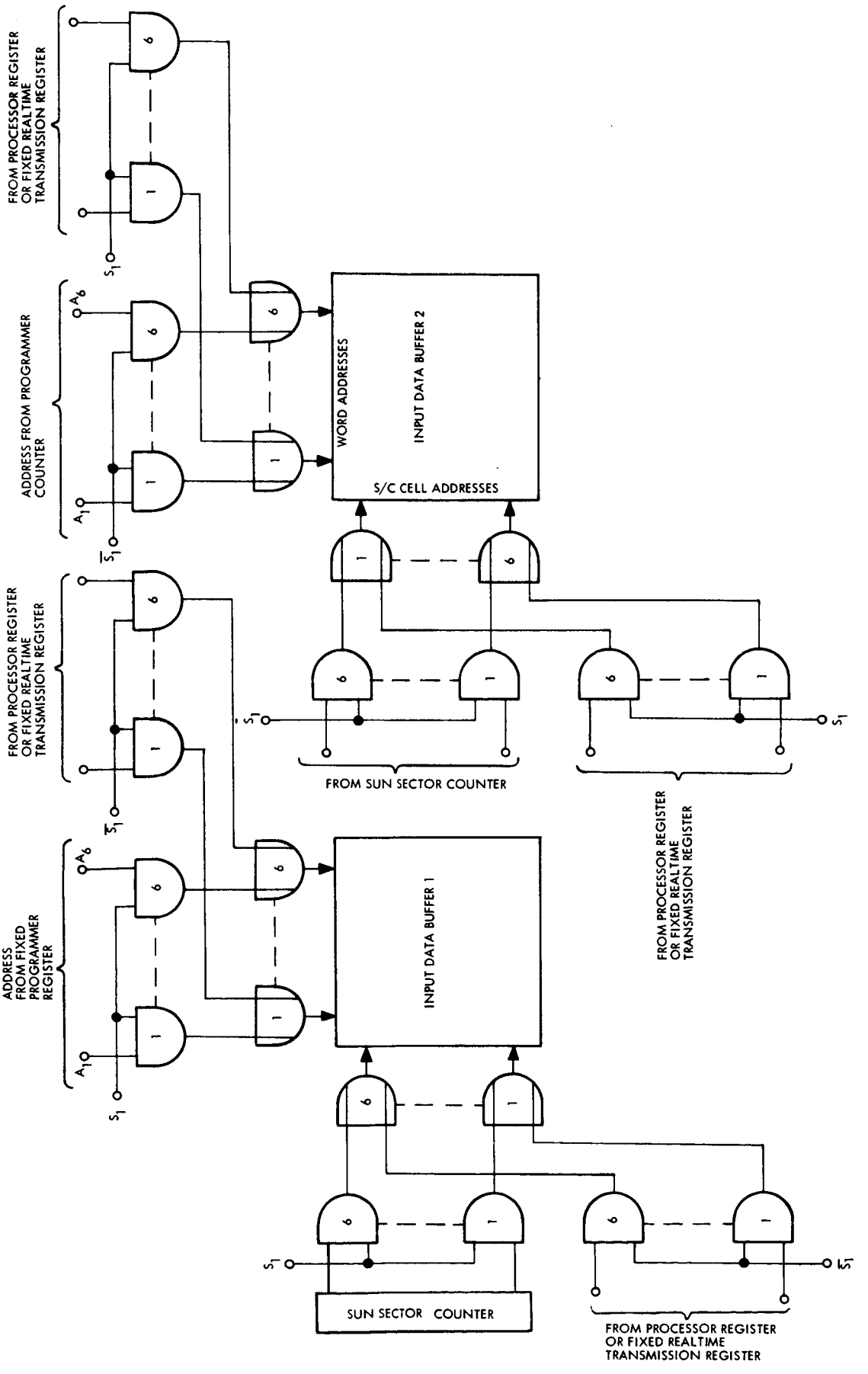


Figure 4-21. IDB Addressing Mechanization

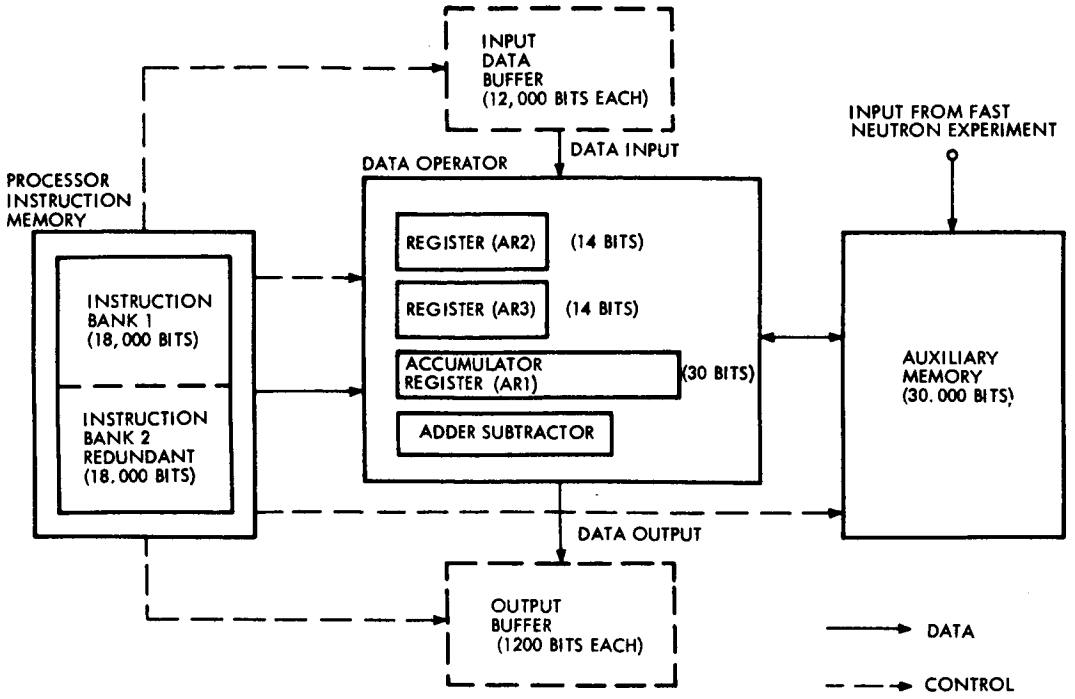
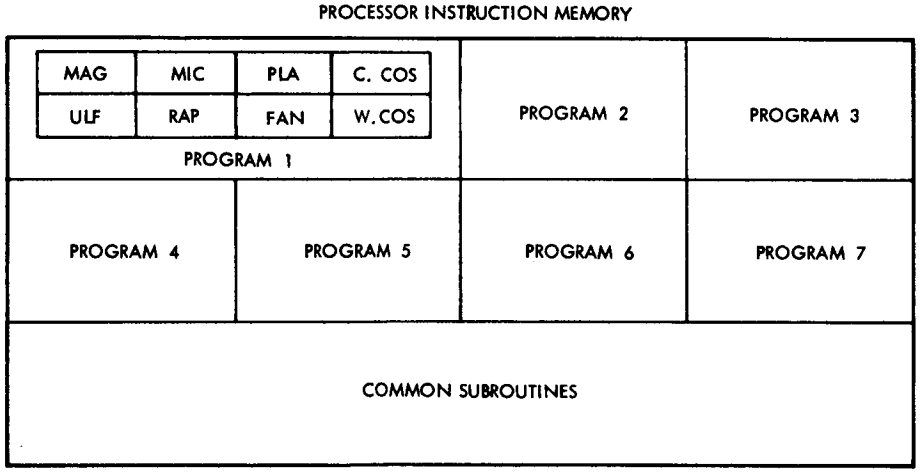


Figure 4-22. Processor Simplified Block Diagram



MAIN PROGRAM 1	TRANSMISSION OF REALTIME
MAIN PROGRAM 2	DATA INTERLEAVED
MAIN PROGRAM 3	WITH STORED DATA
MAIN PROGRAM 4	TRANSMISSION OF REALTIME DATA ONLY
MAIN PROGRAM 5	DATA STORAGE OPERATIONS (BULK STORE)
MAIN PROGRAM 6	TRANSMISSION OF HIGH RATE ENGINEERING DATA ONLY
MAIN PROGRAM 7	SELF TEST

Figure 4-23. Processor Instruction Memory Organization

basic organization of the instruction memory. Each numbered square represents one of the seven main programs. The block labeled "common subroutine" stores those subroutines that are common to various main programs. For example, fetch subroutines, as explained in Section 4.2.3, are common to several main programs. This group of common subroutines will be shared among all of the seven main programs as they are required thereby reducing instruction memory capacity requirements.

The program will enter a specific subroutine by the use of an unconditional jump instruction and will return to the main program by the use of another unconditional jump instruction. The return addresses, however, must be stored at the end of the subroutine by the main program. For example, if a specific subroutine is required by the main program, the main program will store the return address (unconditional jump command) at the end of the subroutine prior to entering the subroutine. The main program will then enter the subroutine, perform the subroutine, and return back to the desired addressed location within the main program. The main program will store the return address at the end of the subroutine by use of the modify instruction command. (Refer to Section 4.2.6 for an explanation of this instruction.)

An execution time logic circuit is used to time each processor operation. For example, if an instruction requires 20 clock pulses to perform its operation, the execution time logic will select the next processor instruction at clock pulse 21. In other words, the execution time logic will select the next processor instruction as soon as the previous instruction is complete. This minimizes the time required to process data. An alternate approach is to allow a time period equivalent to the longest instruction execution time for every instruction. This, however, is wasteful of processing time and would necessitate a higher processor clock rate to perform the required processing within the allotted time period.

The three basic operations, fetching, processing, and storing data will be discussed in the following sections. The required processor instructions for these operations are discussed in Section 4.2.10.

4.2.2 Processor Instruction Word Format

The processor instruction word bits control all processor operations. Each instruction is read out of the memory in a programmed sequence and decoded to provide processor control for all processor operations. Figure 4-24 illustrates the processor instruction bit grouping. As shown, bits 1 through 12 may have several different meanings (auxiliary memory address, input data buffer address, output formatting, and special operation) depending upon the operation code. Bits 13 to 17 are the operation code. The operation code defines the process to be performed. Bit 18 is a parity bit. An odd bit parity count is used to check stored instructions each time an instruction is read out of the processor memory. The result of the parity error condition will be discussed later within this report.

Each processor instruction is uniquely identified by bits 13 through 17, the operation code bits. These bits are decoded by an operations decoder consisting of 32 gates which decode the five operation code bits into 32 signals. These 32 signals control the processor logic to perform all processor operations.

4.2.3 Fetching Operations from the Input Data Buffer

Selection or fetching of data words from the IDB may be performed by random access fetch or by sequential fetch. Random access fetch permits the selection of any data word within the IDB by the use of a separate fetch instruction for each word selected. Sequential fetch permits the selection of up to 64 words by the use of a simple three-instruction subroutine.

4.2.3.1 Fetch Random Access

The block diagram of Figure 4-25 illustrates the mechanization of the fetch random access command group. There are two commands to this group:

- Fetch random access one part word
- Fetch random access two part word.

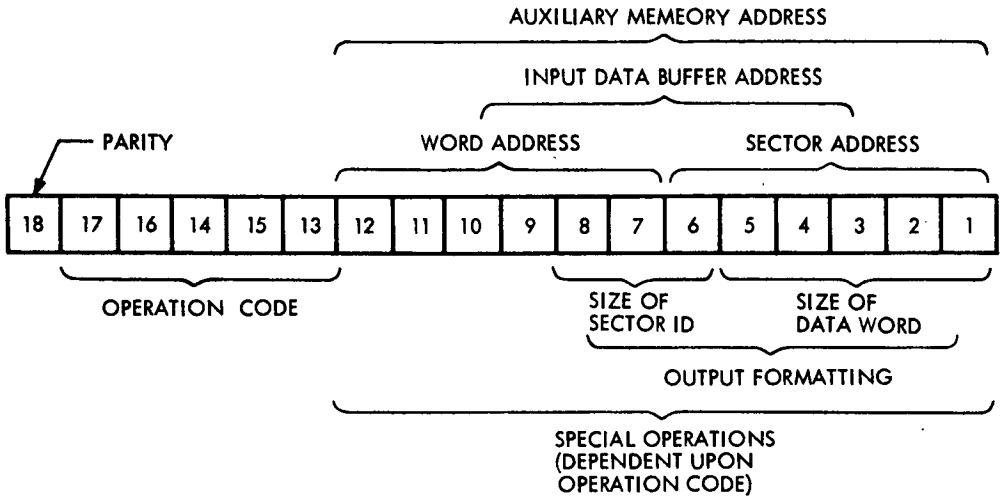


Figure 4-24. Processor Instruction Word Format

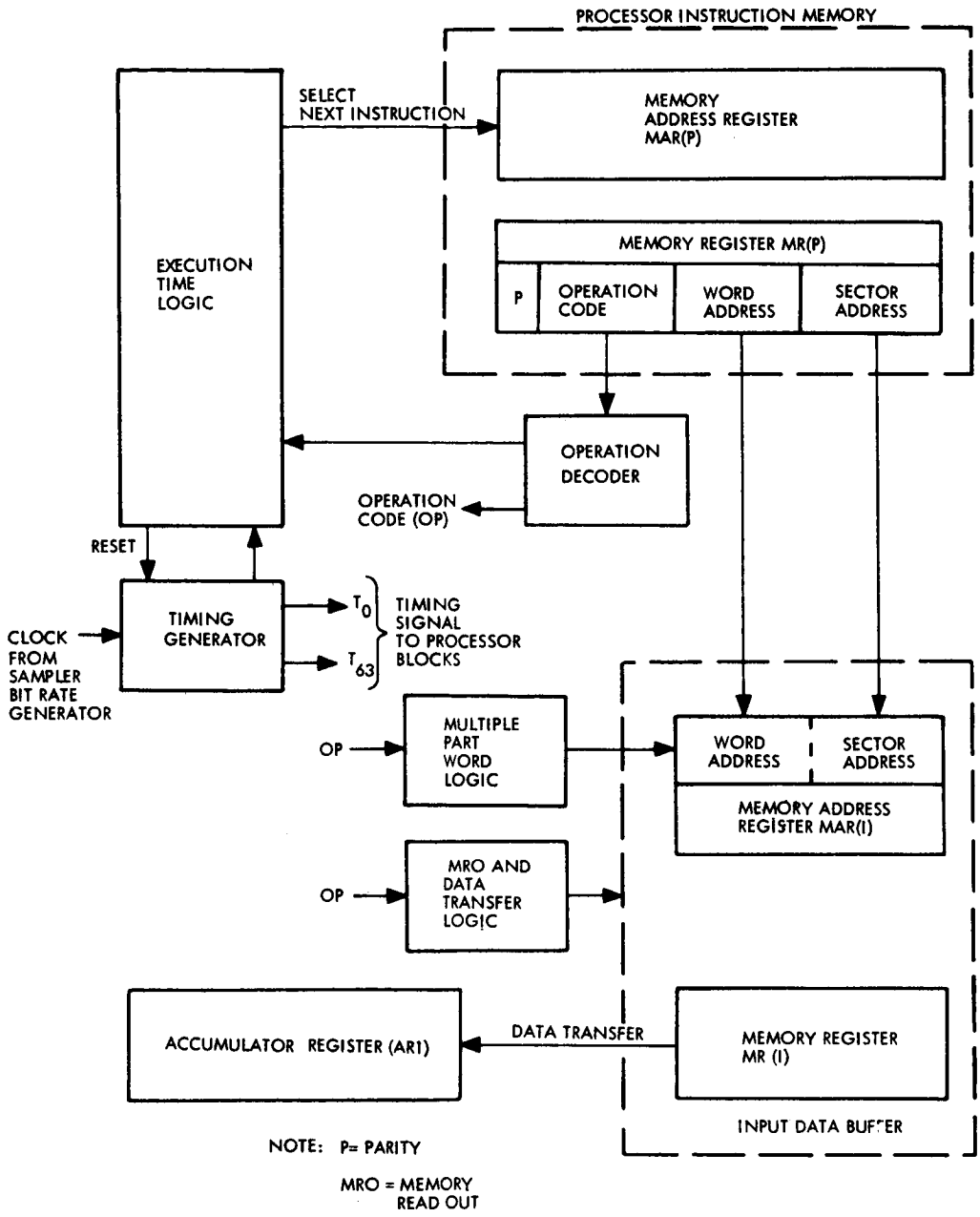


Figure 4-25. Fetch, Random Access Operation Block Diagram

A data sample may contain 16 bits; therefore, two IDB words are required to store the word since the IDB is organized into 8-bit words. This type of word is referred to as a two-part word.

The fetch command and the address of the selected data are contained within the 18-bit structure of the processor instruction word. Data from the input data buffer is serially transferred into the accumulator register in groups of 8 bits. The most significant bit of the selected word appears in the most significant bit portion of the accumulator register. The accumulator register is the main operations register of the processor. That is, addition, subtraction, shifting, etc., take place within the accumulator under the control of processor instruction.

The operation of the block diagram of Figure 4-25 is illustrated by the flow diagram of Figure 4-26. This flow diagram shows the operations and the timing required to fetch a two-part word from the input data buffer.

The timing generator shown in Figure 4-26 generates timing signals at the clock frequency which corresponds to the prevailing downlink transmission bit rate. The processor clock rate versus transmission bit rate is specified as follows:

Processor Clock Rate	Transmission Bit Rate
131.072 kHz	2048, 1024 bps
131.072 kHz	512, 256 bps
65.536 kHz	128, 64 bps
16.384 kHz	32, 16 bps

This timing generator begins counting at the prevailing clock frequency at the onset of each new instruction fetched from the processor instruction memory and is halted by the execution time logic when the instruction sequence has been completed. (Refer to Figure 4-27 for a more detailed block diagram of the processor registers.) The operation code will address either the IDB or the auxiliary memory depending upon the instruction.

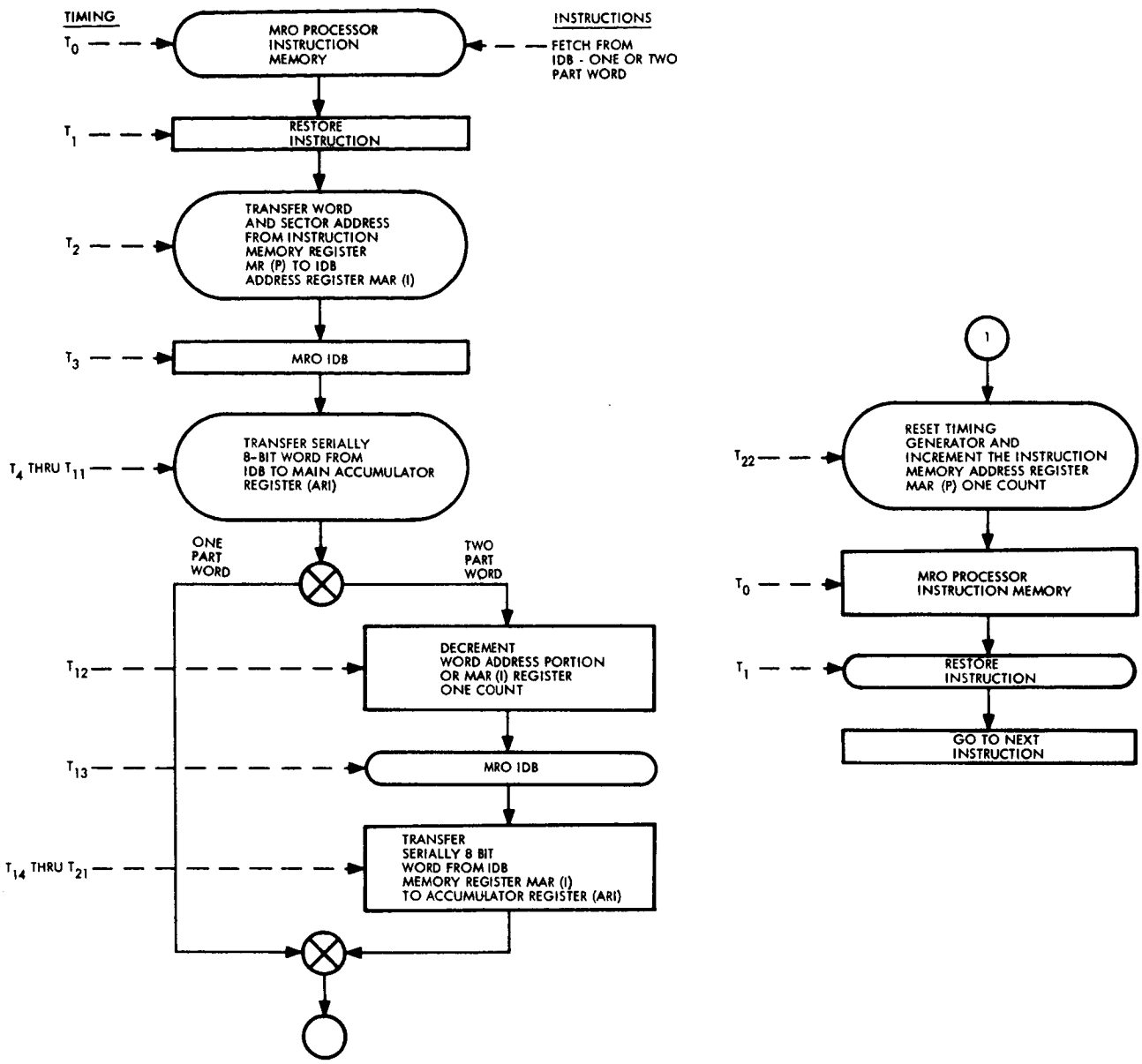


Figure 4-26. Flow Diagram of Fetch Random Access One or Two Part Word from IDB

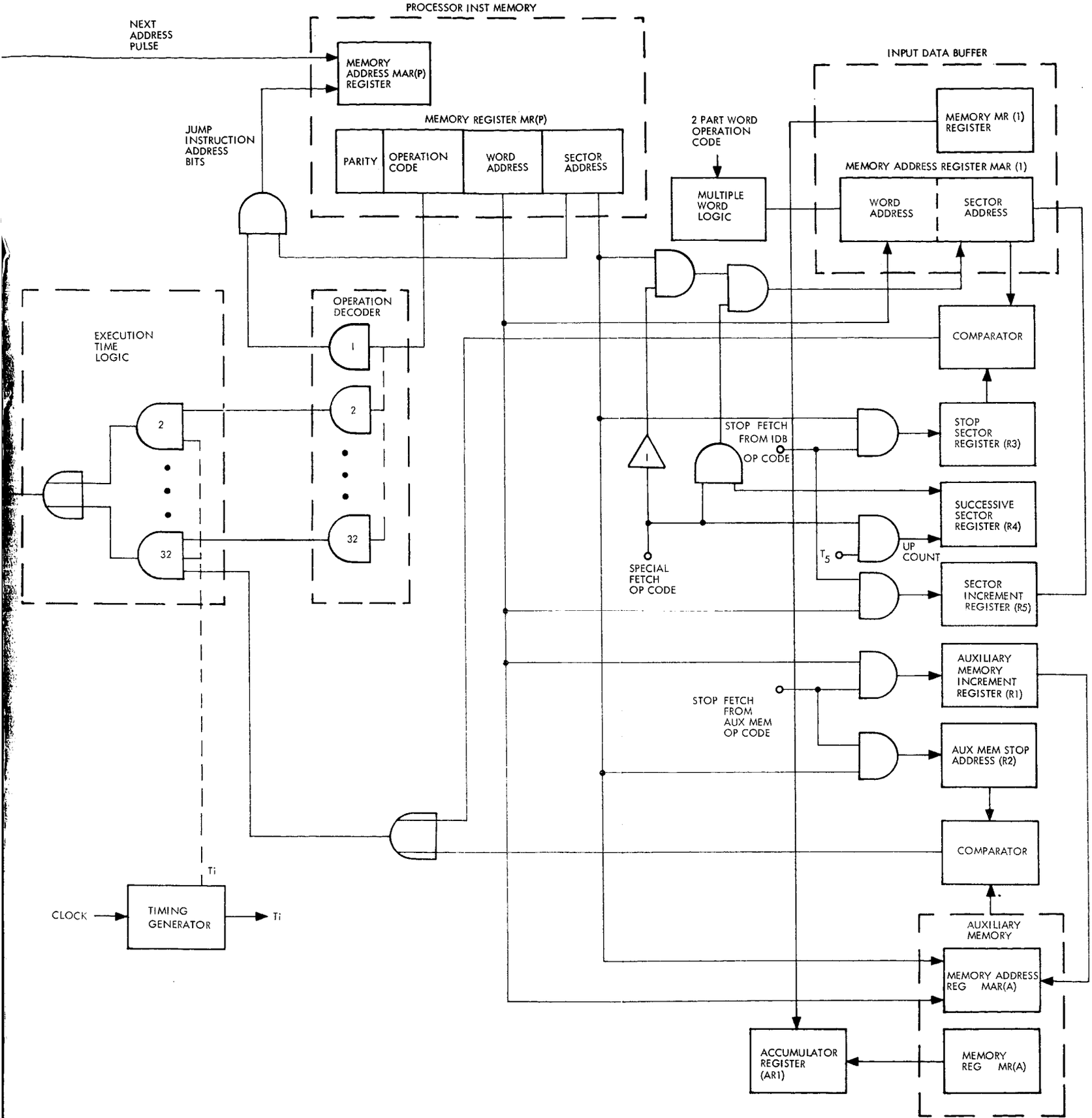


Figure 4-27. Processor Logic Block Diagram

4.2.3.2 Fetch Sequential Subroutine

The fetch sequential subroutine consists of a three-processor instruction subroutine which allows groups of samples to be selected from the IDB in a programmable sequence. This operation can perform the following:

1) Begin the fetching sequence from any one of the 64 IDB sector cells and end on any sector cell. This is achieved by storing the desired start and stop sector addresses in registers R2 and R3, respectively.

2) Increment sectors, that is, select data stored within each of the IDB 64 sector cells, every 2nd, 3rd, 4th, etc. This is achieved by loading an increment code into the sector increment register R5 and updating register R_2 by the amount indicated by R5 as required by the subroutine.

3) Select a one or a two-part word. This is achieved by having two sequential fetch instructions, one for each type word.

The operation of the sequential fetch subroutine is illustrated by Figure 4-28. Figure 4-29 is a block diagram of the processor registers used for this operation.

The instructions required within the subroutine are the following:

1) Fetch random access one-part word from IDB
   Put word address → MAR(I)
   Put sector address → MAR(I).

2) Fetch random access two-part word from IDB
   Put word address → MAR(I)
   Put sector address → MAR(I).

3) Stop address (IDB)
   Put word address → R5 (sector increment register)
   Put sector address → R3 (stop sector address).

4) Test for stop address (IDB).

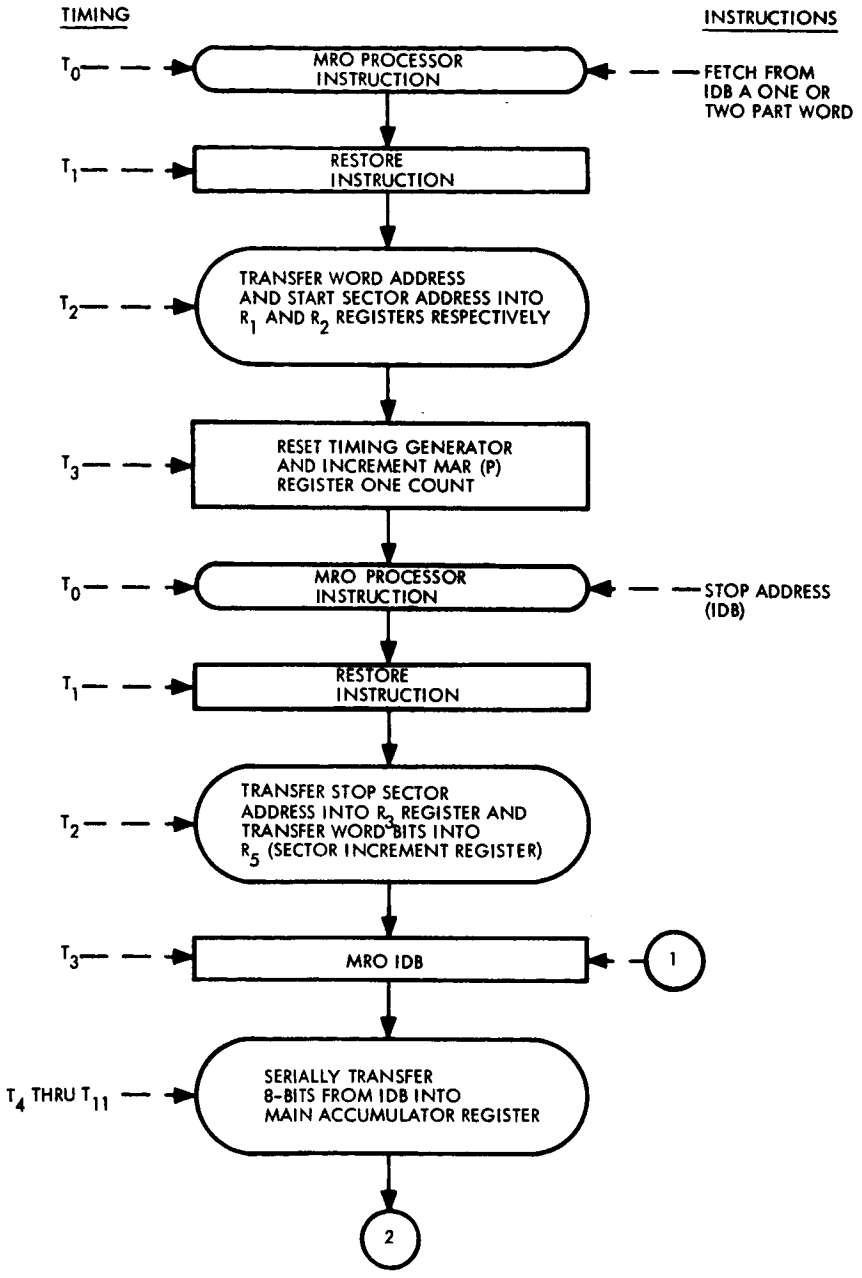
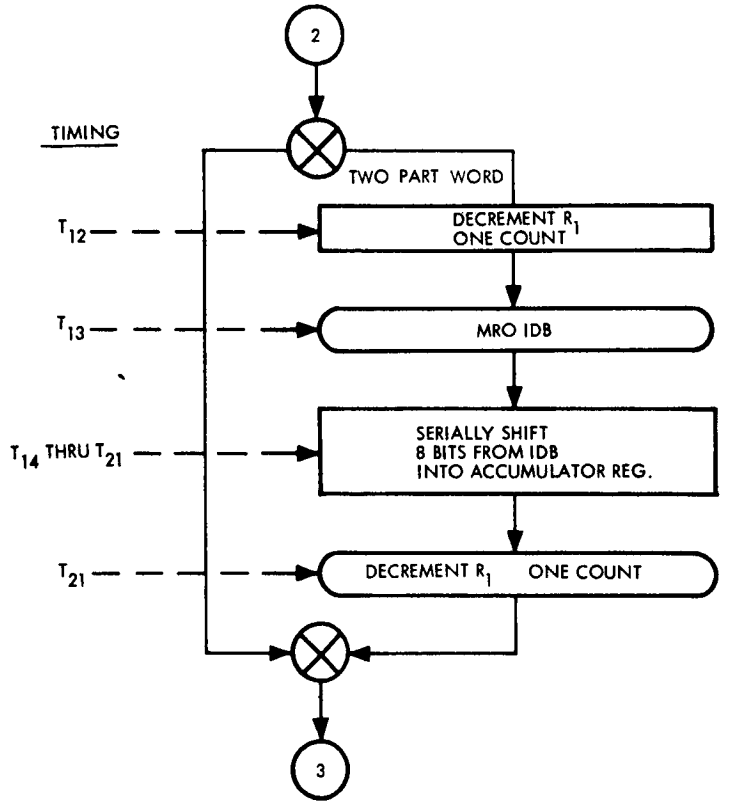


Figure 4-28. Flow Diagram of Sequential Fetch Subroutine (One or Two Part Word from IDB)



MAR (P) =PROCESSOR ADDRESS REGISTER

MR (P) = PROCESSOR MEMORY REGISTER

MAR (J) = IDB SECTOR AND WORD ADDRESS REGISTER.

R3 = STOP SECTOR ADDRESS REGISTER

R5 = SECTOR INCREMENT REGISTER

Figure 4-28 (Continued)

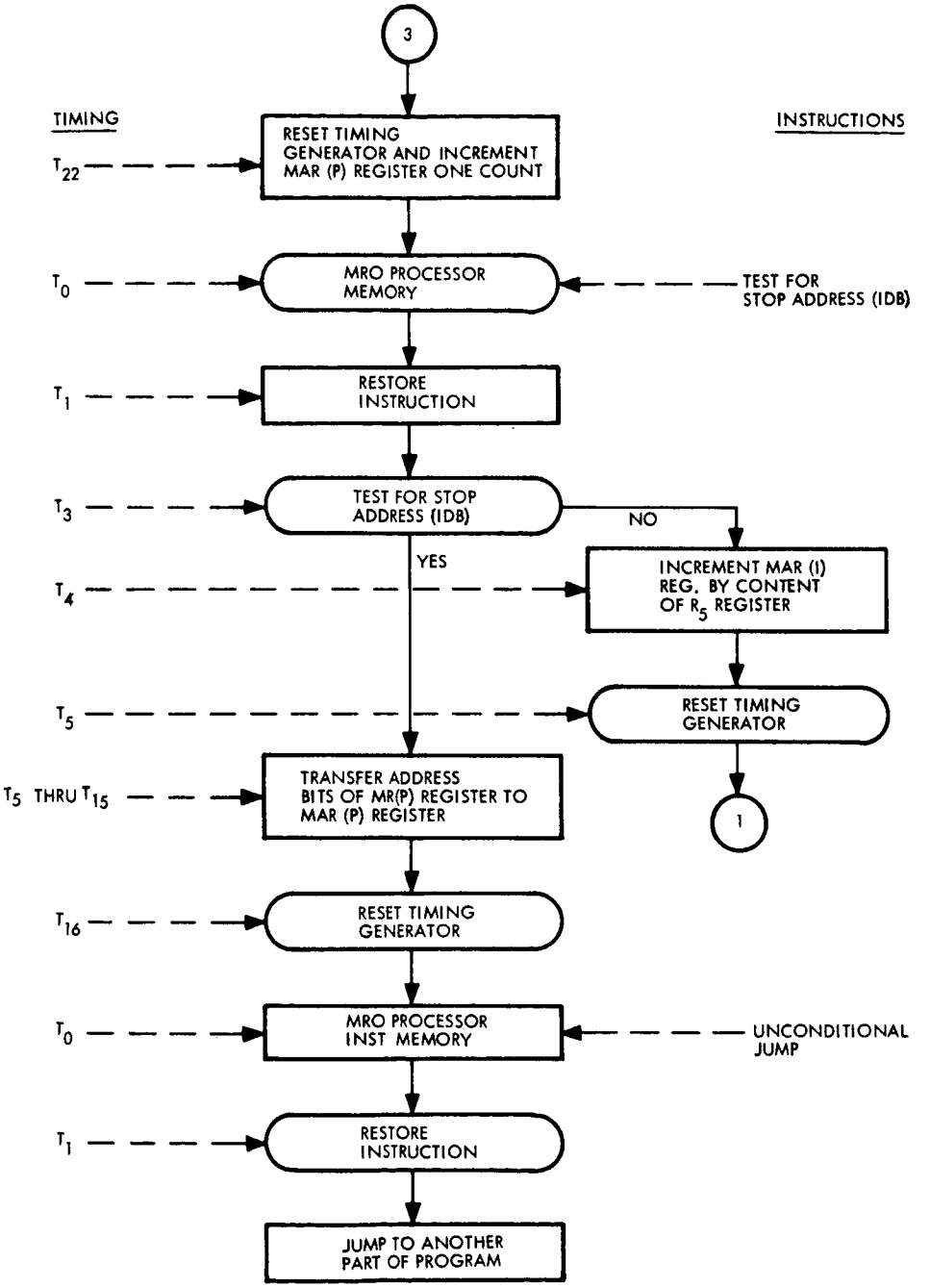


Figure 4-28 (Continued)

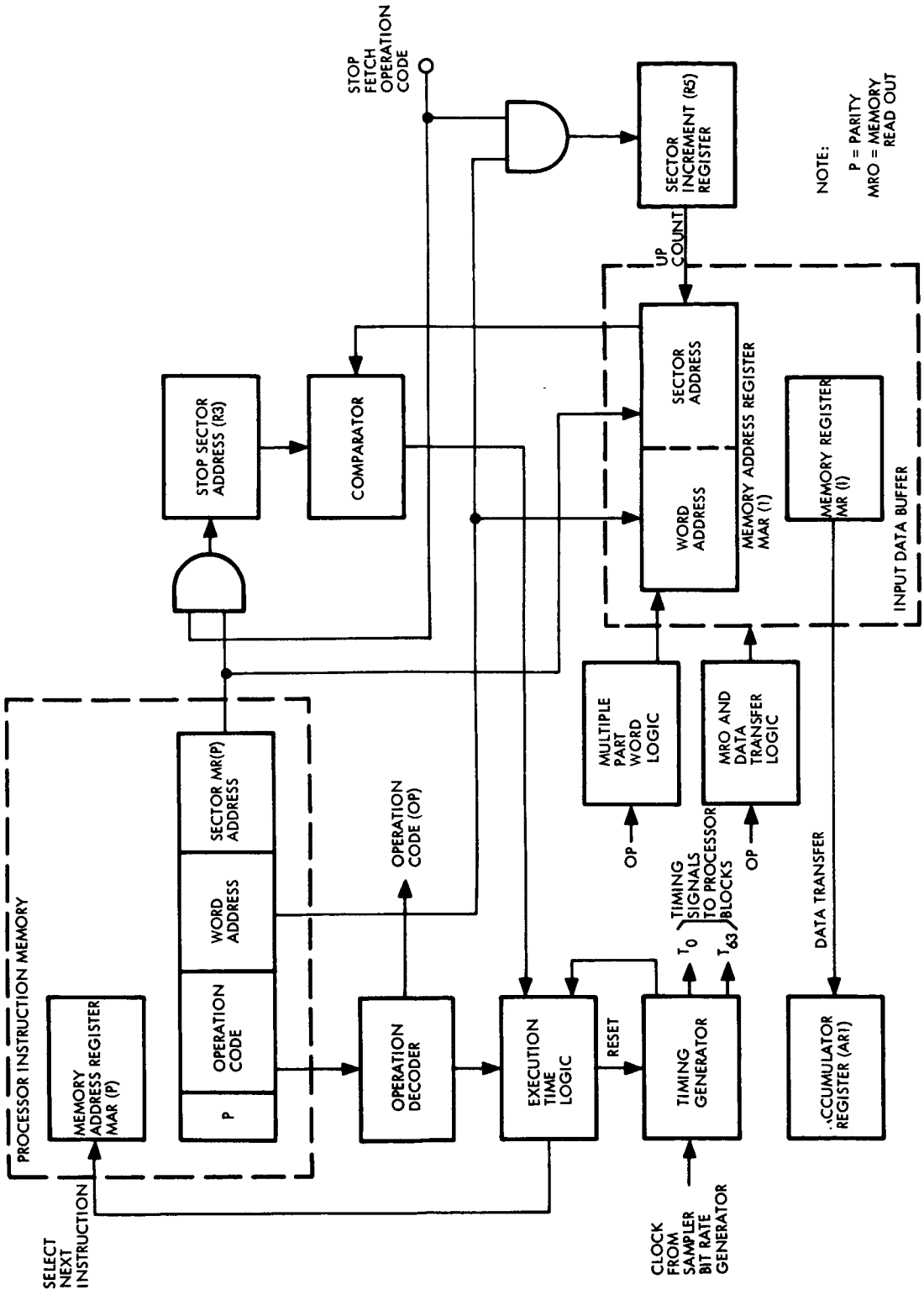


Figure 4-29. Fetch Sequential Subroutine for IDB Block Diagram

Note, the sequential fetch subroutine uses the random access instruction. The random access instruction is used within the subroutine to load the start sector address and the word address into registers R2 and R1, respectively.

4.2.3.3 Special Fetching Operation

Data from certain experiments (e.g., cosmic ray) must be selected from succeeding sectors each spacecraft revolution. For example, if data from sector number 24 is selected during one spacecraft revolution, sector 25 is selected the next revolution, etc. A separate instruction is provided for this type of data selection (the fetch special instruction). One, two, or four samples may be selected each spacecraft revolution depending upon the processor program in use. Where more than one sample is selected per spacecraft revolution, the sectors are sampled in succession. For example, if sectors 1, 2, 3, and 4 are sampled during one revolution, then sectors 5, 6, 7, and 8 will be sampled the next revolution, etc.

Figure 4-27 shows the circuitry required to mechanize this special operation. A successive sector address register R4 is provided which stores the last sector address. When a special fetch operation code is present, the content of R4 is put into the sector address register R2. Register R4 is incremented at T_5 time each time a special fetch operation code is present.

4.2.4 Fetching Operation from the Auxiliary Memory

There will be two basic types of fetching operations from the auxiliary memory: one a random access fetch and the other a sequential fetch operation. The random access fetch operation requires an instruction for each data selected. The sequential fetch operation, however, requires only one fetch instruction for each programmed group of desired samples. Figure 4-27 shows the registers and basic logic for the auxiliary fetching operations.

The flow diagram of Figure 4-30 illustrates the subroutine used to fetch sequentially words from the auxiliary memory. Figure 4-31 illustrates the operations required to fetch random access data from the auxiliary memory. The flow diagrams do not reflect the sharing of

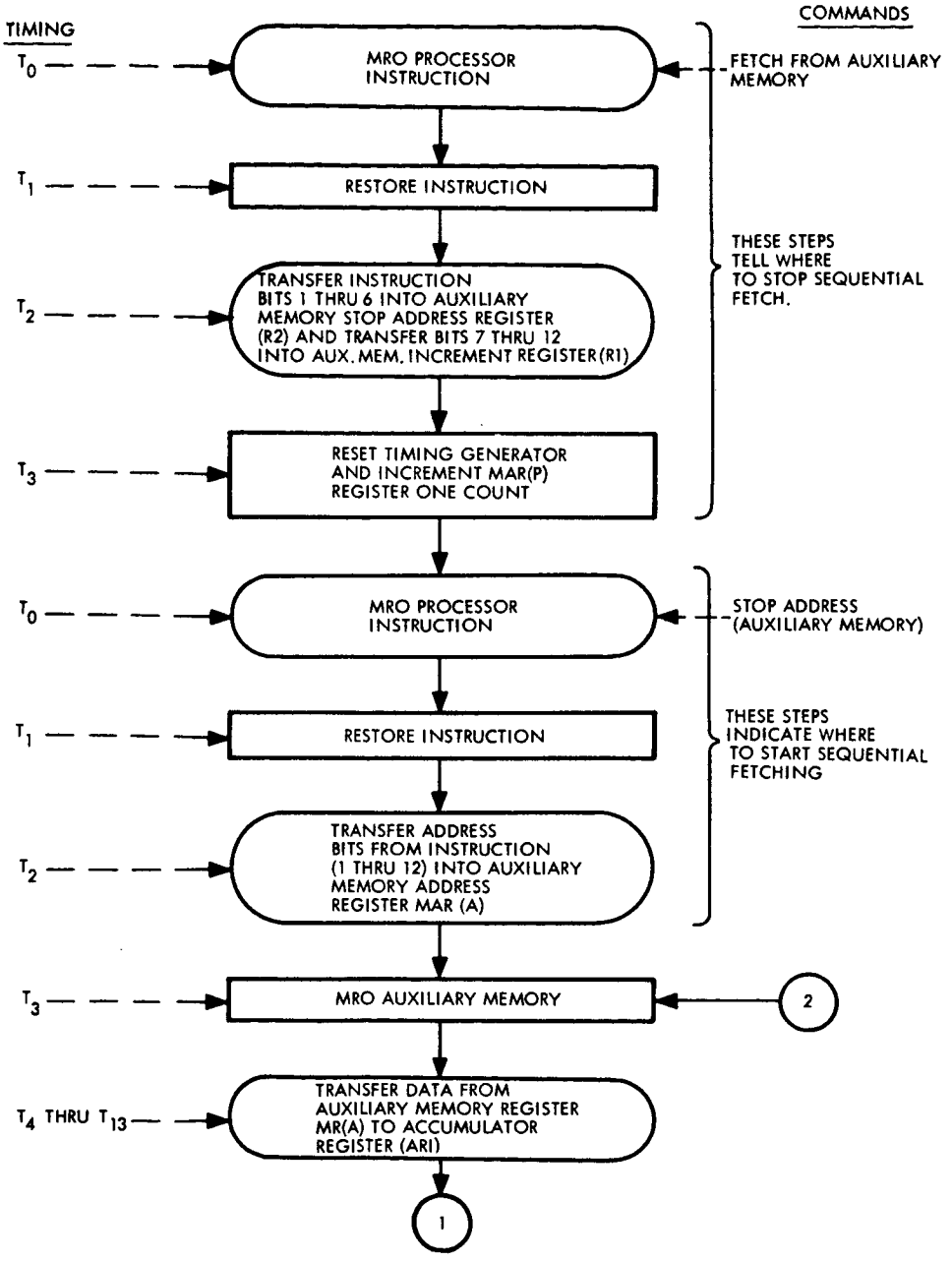


Figure 4-30. Flow Diagram of Fetch, Sequential from Auxiliary Memory Subroutine

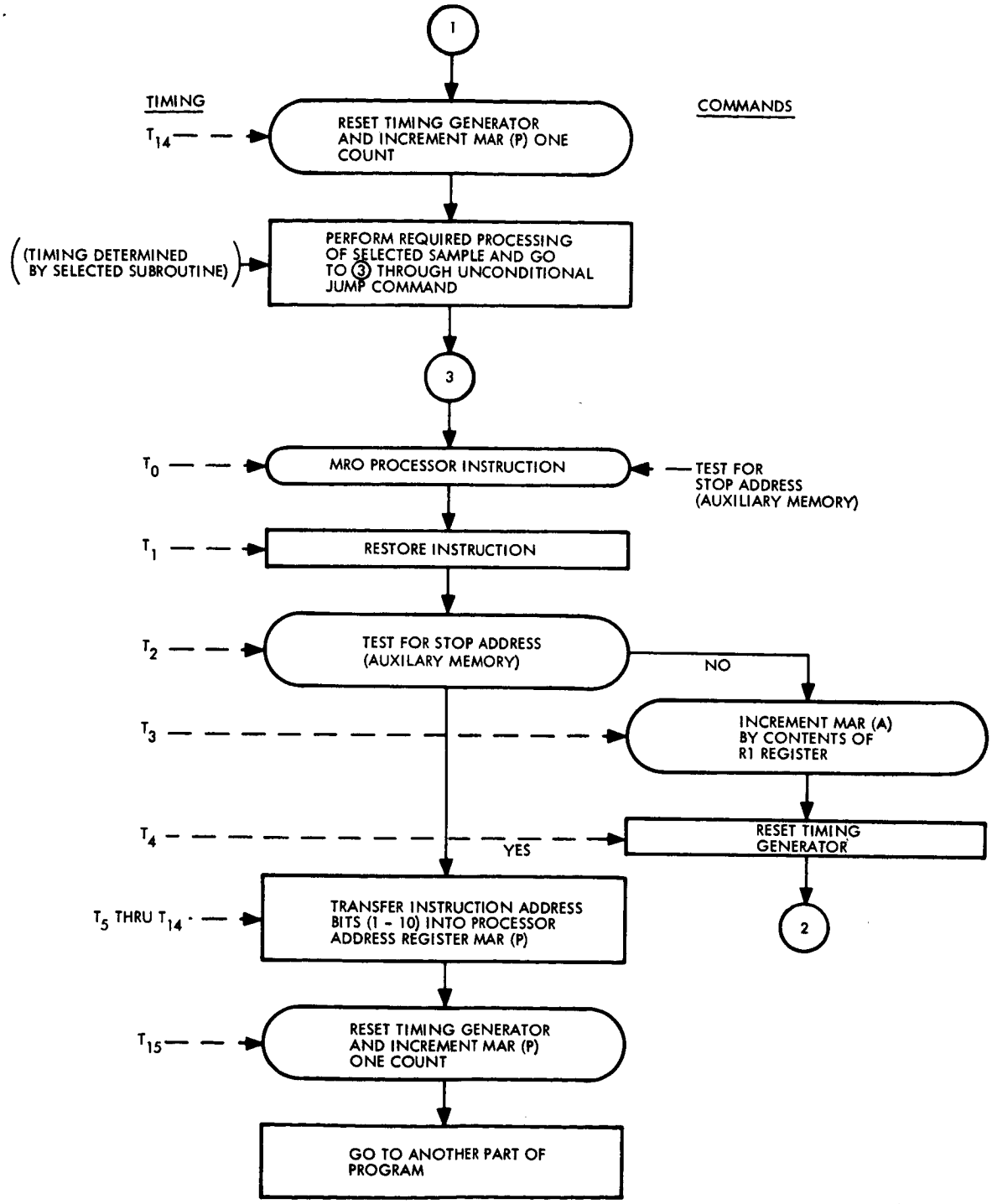


Figure 4-30 (Continued)

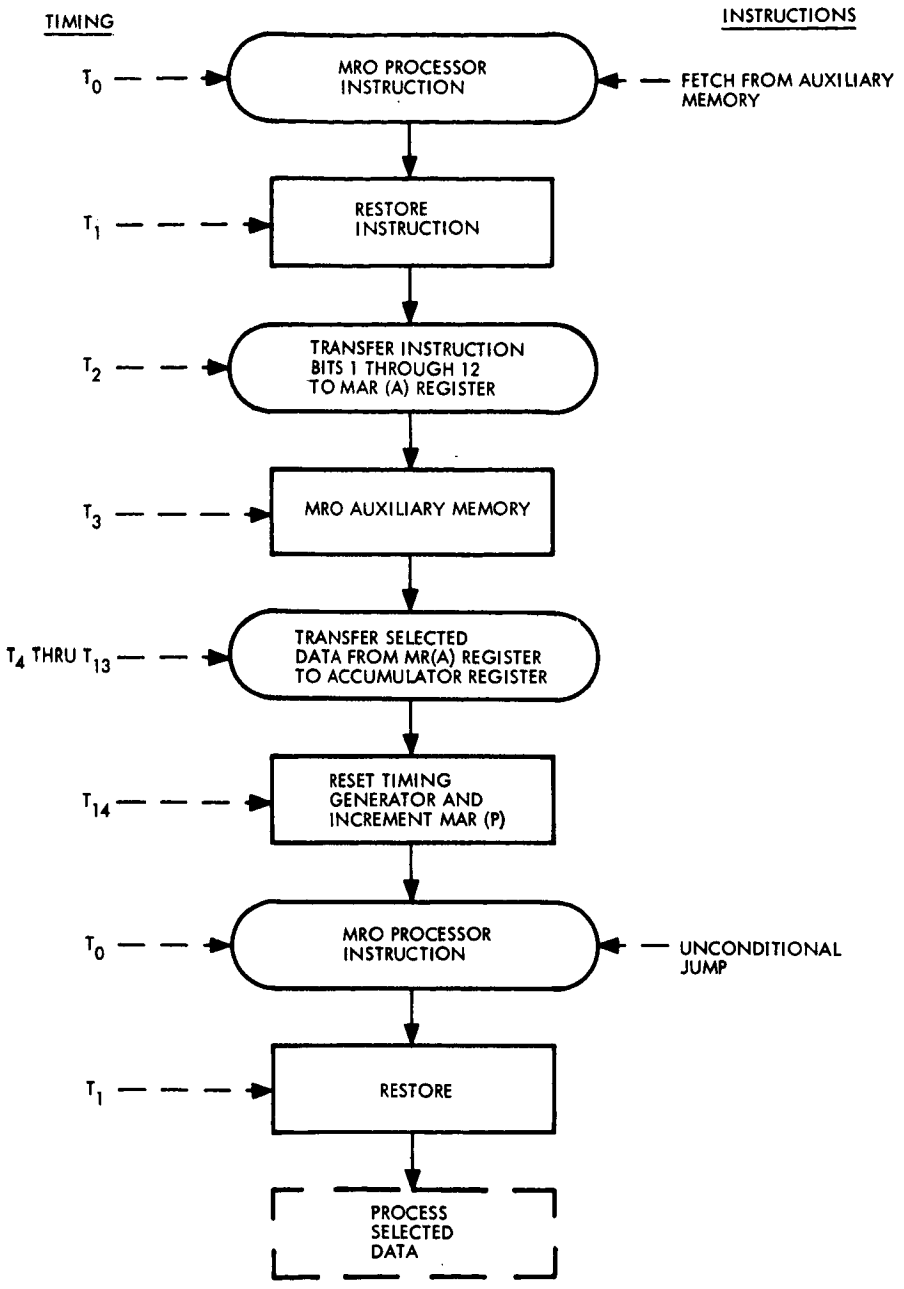


Figure 4-31. Fetch Random Access from Auxiliary Memory

common subroutines. The fetch random access operation requires only one processor instruction called fetch. Random access fetch sequentially, however, requires a subroutine consisting of three basic instructions:

- Fetch from auxiliary memory
- Stop address (auxiliary memory)
- Test for stop sector address (auxiliary memory).

Refer to section 4.2.10 for a discussion of processor instructions.

4.2.5 Data Storage into the Auxiliary Memory

There are two types of store instructions for the auxiliary memory: sequential store and random store. The sequential store operation provides for storing a block of data sequentially by use of one store instruction. The random store operation provides for storage of data into any addressed location. This operation, however, requires a separate instruction for each location. The flow diagram of Figure 4-32 illustrates the random access store operation and Figure 4-33 illustrates the sequential store operation.

4.2.6 Data Storage into Output Buffer

Processed data is stored into the output buffer bit serially from the accumulator register. This process is part of the formatting operation which is controlled by the processor program. The number of shift pulses required is controlled by the processor program and determines the length of the word to be transferred to the output buffer. The sector address, if required, follows the least significant bit of the data to be stored. The number of sector bits required to identify data is programmable.

Each word to be stored into the output buffer from the accumulator register requires a separate instruction. Bits 1 through 5 of the instruction word are used to determine the length of each word to be transferred to the output buffer. Bit positions 6 through 8 determine the length of the required sector ID. The flow diagram of Figure 4-34 illustrates the output buffer store operation. (Refer to Appendix C for the overall CDS data flow diagram.) Transfer of data to the output buffer takes place at the following rates:

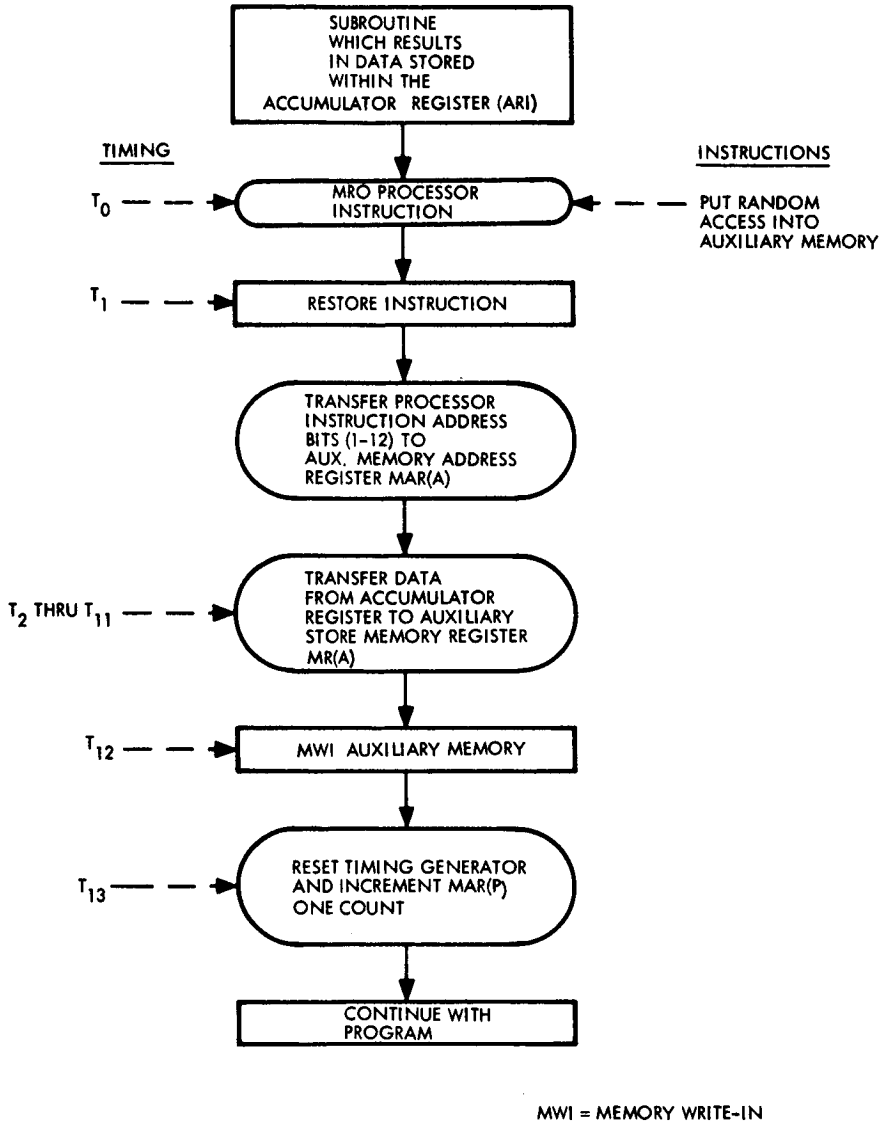


Figure 4-32. Put, Random Access into Auxiliary Memory Flow Diagram

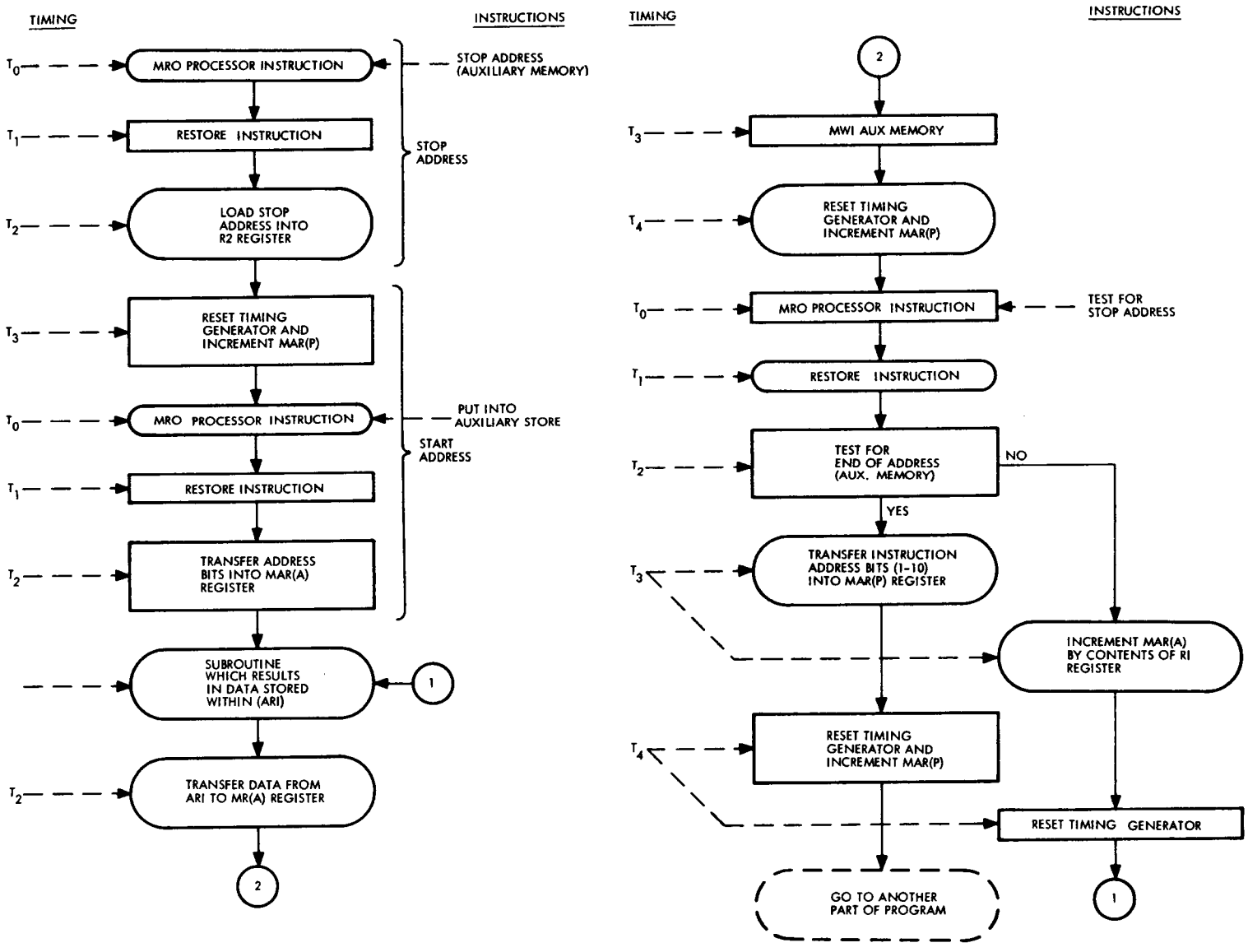


Figure 4-33. Put, Sequentially Subroutine for Auxiliary Memory Flow Diagram

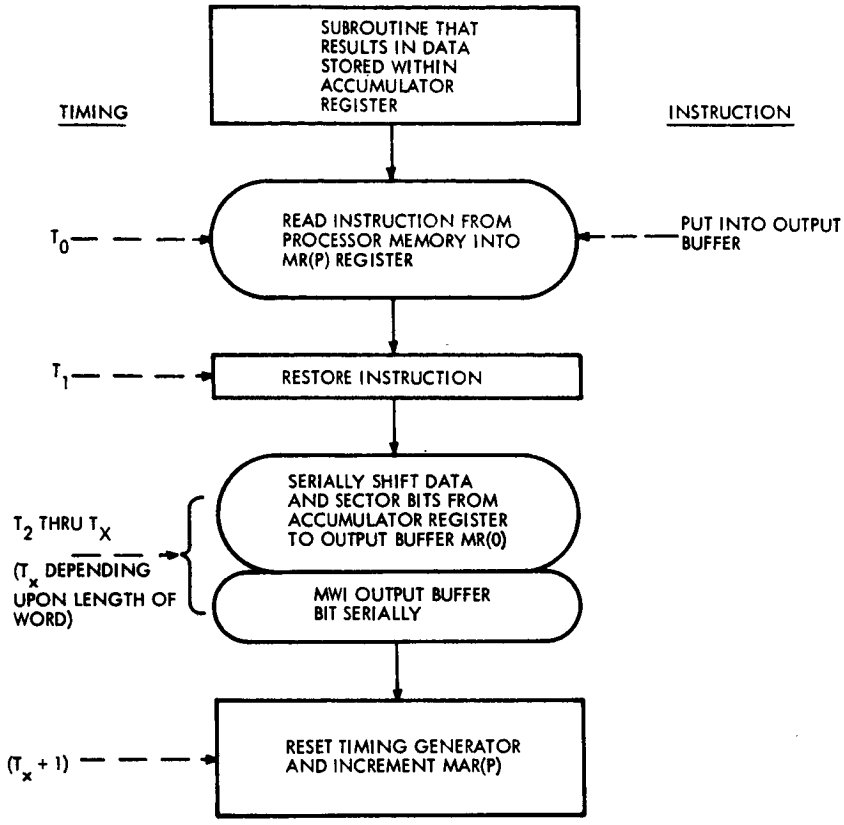


Figure 4-34. Output Buffer Store Operation Flow Diagram

Transmission Bit Rate	Write-In Clock Rate
2048, 1024 bps	2 kHz
512, 256 bps	1 kHz
128, 64 bps	500 Hz
32, 16 bps	100 Hz

4.2.7 Special Processor Operations

The processor is required to perform various special operations such as:

- Check the processor instruction word for parity
- Test for asynchronous event
- Change processor programs.

The following section describes these special operations.

4.2.7.1 Parity Check

Bit position 18 of the processor instruction word (Figure 4-24) is allocated as a parity bit. Each time an instruction is read out from memory it is checked for parity. When a parity error is detected, the CDS automatically enters the fixed realtime engineering format which informs the ground station of a parity error and its memory location. The ground station then may command readout of the entire processor stored program which is in error. When an error is found, the program may then be corrected by ground command. That is, if the error is minor, specific instruction words may be refilled via the command link. The ground station may also command switchover to the redundant processor instruction memory if deemed necessary. During program refilling operations, the CDS transmits in the fixed realtime mode.

4.2.7.2 Selection of Stored Programs

The selection of processor programs is performed by ground command. The address of the first instruction is transmitted to the spacecraft via the command processor. This address is then entered into the processor instruction memory address register MAR(P) to select the desired program sequence. The mechanization is shown in Figure 4-35. Mode of operation changes occur only at the end of transmission frames.

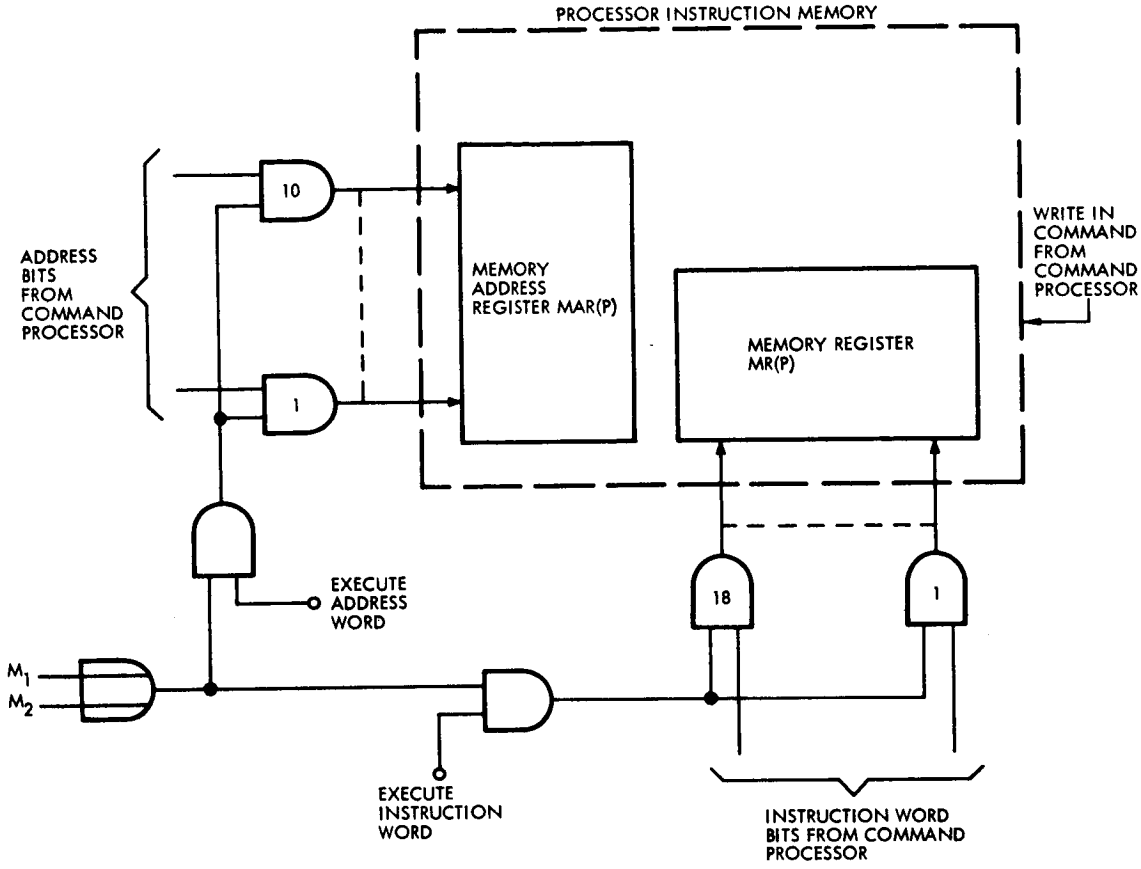


Figure 4-35. Processor Program and Instruction Change

4.2.7.3 Program Changes

The processor instruction words may be changed by ground command. First the address of the instruction to be changed is transmitted. This sets the MAR(P) register to the proper address. Second, the instruction bits are transmitted and entered within the instruction memory. Figure 4-35 illustrates the mechanization for selecting processor programs and changing processor instructions. Processor program changes are performed when the CDS is in fixed realtime modes of operation. The timing generator is inhibited during processor program changes and the command processor provides the required timing pulses to the processor memory. During program changes the processor instruction word which is normally transmitted in the subcomm of the fixed format is inhibited.

4.2.7.4 Asynchronous Data Events

The processor is flagged by the input data sampler when an asynchronous event takes place (micrometeoroid event or fast neutron event). If an event has occurred, a subroutine is called upon to process the asynchronous data. This operation may be considered a conditional jump instruction.

When a micrometeoroid event occurs, the processor receives a flag from the input data sampler. The processor then enters a special routine to process and format this data and enter the processed data within the low rate data portion of the frame. This data will be put in place of very low frequency experiment data within the frame.

The fast neutron experiment requires special handling because of the following points:

1) Special signal conditioning is required (pulse height analysis of neutron energy into ten levels).

2) Irregular sampling periods (sample experiment for 1 second three times per minute for a period of 1 hour each 24 hours during solar activity).

3) The sensor can only sample data for 1 hour during solar activity before sensor saturation.

4) Data rates during the sampling period are high (148 bps).

5) The experiment is only applicable within 0.5 AU of the sun.

6) Data is primarily collected during solar flares. Where no flares occur, very little if any data transmission will be necessary.

The following ground rules are established for the fast neutron experiment:

1) The experiment will flag the CDS when to begin sampling data.

2) Data from the experiment will be sampled three times each minute.

3) Experiment data will be stored in the auxiliary memory continuously for 1 hour at the predetermined intervals indicated above. Data will be stored directly into the auxiliary memory from the input data sampler.

4) Experiment data will be transmitted within the low rate data group (see Section 3.3.2.2).

5) Fast neutron data sampling, processing, and formatting will be under the control of the processor program.

Figure 4-36 shows the implementation of the fast neutron sampling arrangement. Data is collected within 13 registers (FN1-FN13).

The ten counters (FN1-FN10) collect neutron energy levels; FN11 collects charged particle counts, FN12 collects gamma counts, and FN13 collects neutron counts. The processor will serially shift the contents of each of these registers into the auxiliary memory. The shifting and timing operations are under the control of the processor program.

When fast neutron data has been collected, the processor enters a special routine to process and format this data for transmission. During a 1 second sampling period, 148 bits of data are collected. There are three sampling periods each minute; therefore, there are 180 sampling periods in 1 hour. When data is log scaled, the total number of bits is reduced from 148 to 115. Therefore, the number of bits collected and processed after 1 hour is equal to 180 samples x 115 bits/sample = 20,700

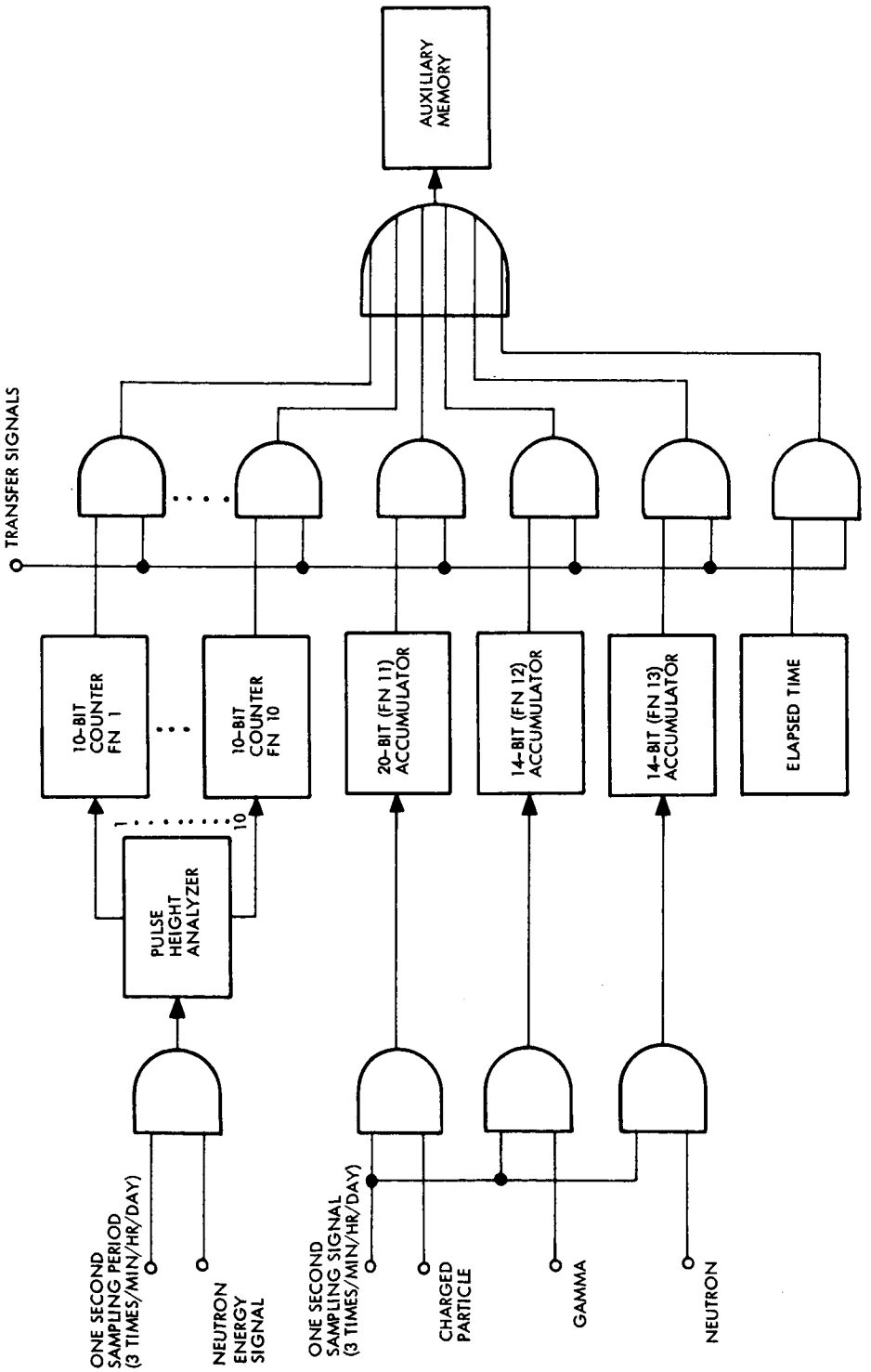


Figure 4-36. Fast Neutron Sampling and Data Transfer Block Diagram

bits. The elapsed time of the onset of the event is recorded and transmitted. Fast neutron data is transmitted in 34-bit blocks within the low rate data group. One 34-bit group is transmitted each 4 seconds; therefore, the 20,700-bit group is transmitted over a period of 1 hour at the highest transmission bit rate. At the lower bit rates the transmission period will be proportionately longer.

4.2.8 Arithmetic Operations

Addition and subtraction is performed bit serially within the CDS. Figure 4-37 shows the basic mechanization of the add-subtract operation. Subtraction will be limited to 8 bits; the subtrahend will occupy bit positions P17 through P24 of the main accumulator register. The P24 bit is the most significant bit of the subtrahend. As the subtraction takes place, the contents of A1 are recirculated and the difference is stored in A3.

For addition, the addend occupies bit positions P9 through P24 of the accumulator. The least significant bit of the addend appears in bit position P9. The sum appears in bit position P9 (LSB) through P24 when the addition is completed.

4.2.9 Processor Algorithms

Presently, three basic data processing algorithms are planned for the CDS. These are log scaling, averaging, and minimum-maximum value determination. Table 4-1 illustrates the possible processing algorithms for each experiment.

4.2.9.1 Log Scaling

Log scaling is used to reduce the number of bits required to transmit accumulated pulse data counts. The method involves two basic steps:

1) Count the number of most significant consecutive zeros and record this number using either 2 or 3 bits depending on the size of the original word.

2) Drop off the required number of least significant bits to formulate the log scaled word. Figure 4-38 illustrates the mechanization for log scaling a 20-bit word.

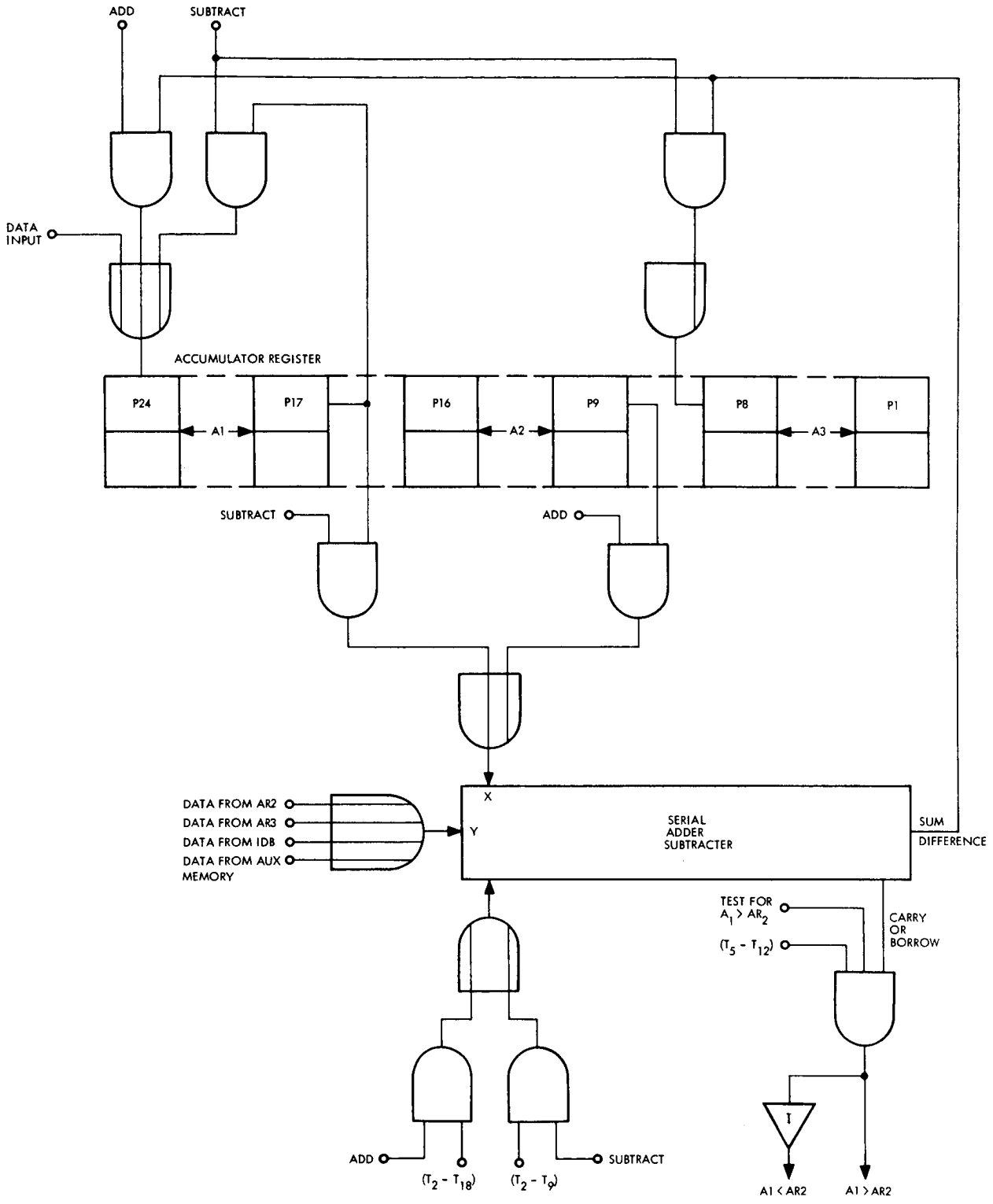


Figure 4-37. Logic Operation of Addition and Subtraction

Table 4-1. Experiment Processing Algorithms and Data Bits

Experiment*	2048 1024 Bit Rates		512 256 Bit Rates 128		64 32 Bit Rates 76	
	Algorithms	Bits Per Frame	Algorithms	Bits Per Frame	Algorithms	Bits Per Frame
Magnetometer	Select eight samples per rev	192	Select four samples per rev	96	Select one sample per rev	24
Chicago cosmic ray	Select four samples per rev +16 sector ID bits. Sample subsequent sectors each sampling period.	312	Select two samples per rev +8 sector ID bits. Sample subsequent sectors each sampling period.	164	Select one sample per rev log scale, 4 sector ID bits required. Sample subsequent sectors each sampling period.	65
Plasma probe	Select 16 samples per rev +13 bits for max flux	125	Select four samples per S/C rev +13 bits for max flux	41	Select one sample. Determine max value of flux	20
Radio propagation	Select 16 samples per sec	112	Calculate min-max value of S_5	24	Calculate min-max value of S_5	24
Very low frequency	Select one sample (X_c, Y_c, Z_c)	24	Select one sample (X_c, Y_c, Z_c)	18	Select one sample (X_c, Y_c, Z_c)	18
Webber cosmic ray	Select four samples per rev, log scale, trans 16 sector bits. Sample subsequent sectors each sampling period.	213	Select two samples per rev, log scale, trans 8 sector bits. Sample subsequent sectors each sampling period.	156	Select one sample per rev, log scale, trans 4 sector bits. Sample subsequent sectors each sampling period.	78
Micrometeoroid	Transmit 34 bits only when event occurs in low rate data group	34**	Transmit 34 bits only when event occurs in low rate data group	34**	Transmit 34 bits only when event occurs in low rate data group	34**
Fast neutron	Collect data for one hour 20,700 bits and transmit in 34-bit blocks in low rate data group.	34**	Collect data for one hour 20,700 bits and transmit in 34-bit blocks in low rate data group.	34**	Collect data for one hour 20,700 bits and transmit in 34-bit blocks in low rate data group.	34**
Low rate data	See Figure 3-10	64	See Figure 3-10	64	See Figure 3-10	64
Subcomm	See Table 3-2	8	See Table 3-2	8	See Table 3-2	8
TOTAL		1050		571		301

*Refer to Research Report No. 2 for details on experiment processing requirements.

**These data bits contained within 64 low rate data bits.

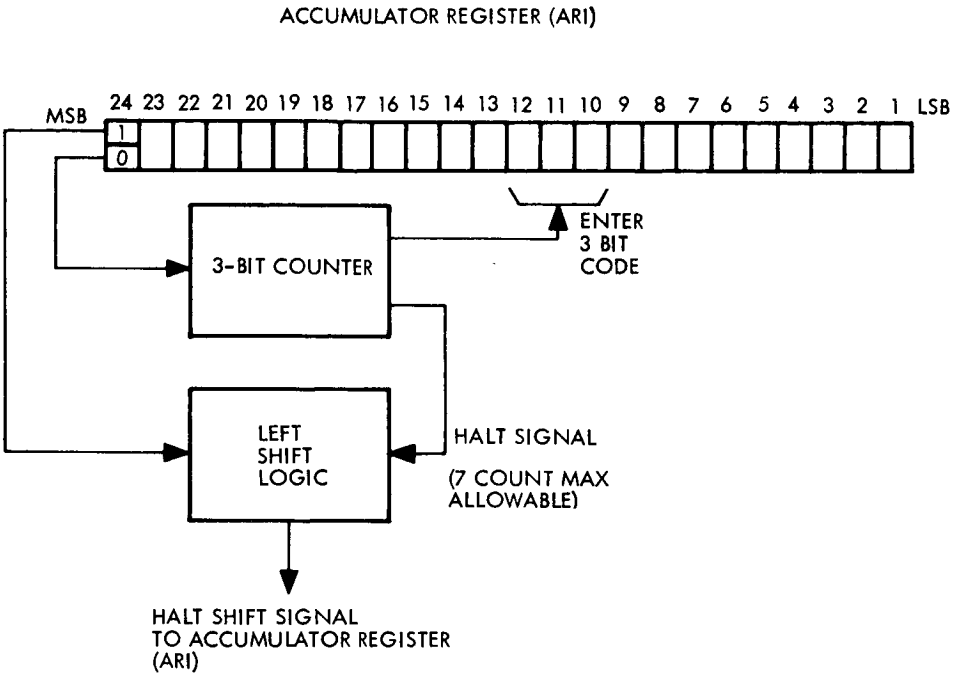


Figure 4-38. Log Scaling Block Diagram for 20-Bit Word

The 20-bit word is transferred into the main accumulator register. The scaling command is fetched from the processor memory. A left shift occurs each clock pulse when the 24th bit is a zero. A count is added to the 3-bit counter for each zero detected. Shifting stops at the first "1" detected in the 24th bit position. The accumulated count, in the 3-bit counter, is entered into bit positions 10, 11, and 12. These three bits indicate the number of most significant, consecutive zeros contained within the original number. The resultant log scaled 20-bit word now consists of 15 bits, i.e., bits 10 through 24 of the main accumulator register.

Table 4-2 summarizes the bit savings and resultant worst case error of six typical number sizes. As shown in the table, a 20-bit word becomes a 15-bit word after log scaling. The worst case error for a 20-bit word is 0.0064 percent. This is calculated assuming the worst case condition where no zeros are detected. In this case, 8 of the 20 bits are virtually dropped; therefore, $2^8 = 256$ of the $2^{20} = 1,048,576$ counts are lost. This is approximately 0.026 percent.

Table 4-2. Bit Saving by Log Scaling and Resultant Accuracy

Number of Bits in Original Word	Number of Bits in Log Scaled Word	Number of Designator Bits *	Bit Saving	Worst Case Resultant Error
20	15	3	5	0.026%
18	13	3	5	0.098%
16	11	3	5	0.39%
14	9	3	5	1.56%
12	8	3	4	1.60%
10	8	2	2	1.60%

*Designator bits represent the bits required to specify how many zeros were detected in the original number.

The log scaling operation requires only one processor instruction.

4.2.9.2 Averaging

The processor has the capability of averaging experiment data on a revolution basis or over many revolutions. The auxiliary memory is used to store accumulated averages where averaging is performed on many revolutions of data.

To minimize the complexity of the averaging mechanization, a careful choice of the number of average samples will be made. The groups will be restricted to 2, 4, 8, 16, etc., samples. This choice allows division by simply shifting right 1, 2, 3, 4, etc., bits.

The following subroutine is used to calculate the average value of a number of data samples (S_i, i = 1 to n):

- Fetch S_1 into main accumulator register A1
- Fetch add A1 + S_2 (result remains in A1)

  .
  .
  .

- Fetch add A1 + S_n
- Shift right ($\log_2$ n times) the average value is now in A1.

4.2.9.3 Minimum-Maximum

Calculating the minimum and maximum values of a group of samples is very useful in reducing data bandwidth. Normally, a rather large number of instruction words and additional arithmetic hardware are required which implies that temporary storage is required within the data operator. (Note -- the IDB read cycle is destructive.) The following approach was chosen in an attempt to minimize memory instruction locations and arithmetic mechanization hardware.

The temporary storage mentioned above consists of two 8-bit registers plus six sector bits. The designations are AR2 for storing the calculated maximum value and AR3 for storing the calculated minimum value. These two registers are also used during the determination of proper scale and offset for the magnetometer experiment and other processor operations. This is accomplished by comparing the encoded data against stored limits.

The following subroutine is used to determine the minimum and maximum values of a number of data samples (S_i, i = 1 to n):

- Sequential fetch S_1
- Put A1 into AR2 and AR3
- Sequential fetch S_2
- Subtract, AR2 - A1
- Test for A1 < AR2
  - If A1 > AR2, put A1 into AR2 and go to address specified by address bits of test instruction
  - If A1 $\leq$ AR2, go to next instruction
- Subtract, AR3 - A1
- Test for A1 > AR3
  - If A1 < AR3, put A1 into AR3 and go to address specified by address bits of test instruction
  - If A1 $\geq$ AR3, go to next instruction
- Test for stop address (end of subroutine)
  - If yes, go to address specified by address bits of test instruction
  - If no, go to next instruction
- Return to step 3 (unconditional jump command).

Upon completion of this subroutine, the maximum value appears in the AR2 register and the minimum value appears in the AR3 register.

When calculating the min-max values of samples for one spacecraft revolution, all data samples are processed. That is, all 64 samples from one data point are used to calculate min-max values.

4.2.10 Processor Instructions

The instructions used by the processor to perform all required processing are listed in Table 4-3.

Table 4-3. Processor Instructions

1)	Add	17)	Stop address (auxiliary memory)
2)	Subtract	18)	Test for stop address (auxiliary memory)
3)	Unconditional jump	19)	Fetch from auxiliary memory
4)	Fetch from IDB a one part word	20)	Put into auxiliary memory
5)	Fetch from IDB a two part word	21)	Fetch add (AR1 + addressed word)
6)	Shift right	22)	Put A1 → AR2
7)	Shift left	23)	Put A1 → AR3
8)	Test for A1 < AR2	24)	Fetch AR2
9)	Test for A1 > AR3	25)	Fetch AR3
10)	Put A1 into AR2 and AR3	26)	Test for asynchronous event
11)	Put contents of AR1 into output buffer	27)	Return address instruction
		28)	Modify instruction
12)	Log scale	29)	Spares
13)	Stop address (IDB)	30)	
14)	Fetch special	31)	
15)	Test for stop address (IDB)	32)	
16)	Put end of block address into (R6)		

1) Add - The add command initiates a serial bit-by-bit add cycle which requires 16 clock pulses. The contents of auxiliary registers AR2 or AR3 are added to the accumulator register AR1. The accumulator register can contain either the addend or the augend.

2) Subtract - The subtract command initiates a serial bit-by-bit subtract cycle requiring eight clock pulses. The subtrahend is obtained from A1 of the accumulator register and is recirculated during subtraction. The minuend is obtained from AR2 or AR3. The difference is placed into A3 of the accumulator register. Refer to Figure 4-37.

3) Unconditional jump - This instruction provides a means to a different part of the program. The jump is achieved by transferring the contents of the address bits of the processor instruction (bits 1 through 10) into the processor address register. This instruction is used at the end of a subroutine to return to the main body of the program.

4) Fetch from IDB a one-part word - This instruction is used to select an 8-bit word from the IDB, random access.

5) Fetch from IDB a two-part word - This instruction is used to select a two-part word from the IDB, random access.

6) Shift right - This command causes the accumulator register (AR1) to be shifted right by the number of bits indicated by bits 1 through 5 of the processor instruction word.

7) Shift left - Same operation as shift right command except the accumulator register is shifted left. These commands are used in log scaling, shifting data into and out of the accumulator register, binary multiplication, division, etc.

8) Test for A1 < AR2 - This test is used during the calculation of min-max. If the condition is true, the processor proceeds to the instruction specified by bits 1 through 10 of the test instruction. If the condition is false, the processor

proceeds with the next instruction. If A1 = AR2 the processor aiso proceeds to the next instruction.

9) Test for A1 > AR3 - This instruction operates identically to instruction 8) except that the test A1 > AR3 is performed.

10) Put A1 into AR2 and AR3 - This command serially shifts the contents of A1 into AR2 and AR3. This command is used for calculation of min-max.

11) Put contents of AR1 into output buffer - This command is used during formatting operation. The number of bits serially shifted into the output buffer is determined by bits 1 through 5 of the processor instruction word. Refer to Figure 4-24.

12) Log scale - This command log scales the number in the accumulator register. Bits 1 through 4 of the processor instruction identify the scale to be used (20-bit word, 18, 16, 14, 12, or 10-bit word). Refer to Section 4.2.9.

13) Stop fetch from IDB - This command is used to load a stop address into the stop sector address register R3 and the sector increment bits into the sector increment register R5. This command is used in the sequential fetching of data from the IDB. Refer to Figure 4-28.

14) Fetch special - This command is used to fetch samples from successive sectors. For example, this command is used when sector 1 is to be sampled during spacecraft revolution 1, sector 2 during revolution 2, etc.
Bits 1 through 6 (IDB sector address bits) of the processor instruction are not used in this instruction. The contents of the successive sector register (R4) is used to address the IDB sector cells. Bits 7 through 12 of the processor instruction are used to address the IDB word cell as is normally done for fetch instructions. Note that register R4 is upcounted each time the fetch special instruction is used. This provides for the selection of succeeding sectors.

15) Put end of block address into R6 - This instruction is used to load the end of frame or end of block output buffer address into the end of frame register R6. This register indicates the size of the transmission frame from an output buffer and also generates marker signals used in CDS timing. Refer to Figure 4-41.

16) Stop sequence in auxiliary memory - This instruction shifts processor instruction word bits 1 through 6 into the auxiliary memory stop address register R2; this indicates where the fetching or putting sequence is to stop. Also, processor instruction word bits 7 through 12 are shifted into the auxiliary memory address increment register. This register indicates the number of locations which are to be skipped between selected samples during fetching operations. This is used when it is desired to select only the 2nd, 3rd, 4th, 5th, etc., sample or auxiliary memory location.

17) Test for stop address (auxiliary memory) - This instruction is provided as a means of indicating when the desired samples are either put into or taken out of the auxiliary memory. For example, if a block of 25 samples is desired from the auxiliary memory, the memory is cycled until the stop address is reached (in this case the stop address is 25 locations after the start address).

18) Fetch from auxiliary memory - This instruction causes the address bits from the processor instruction (bits 1 through 12) to be transferred into the auxiliary memory address register MAR(A). These bits address the desired auxiliary memory word cell. In the case of sequential fetch, these bits specify the location where sequential fetching is to begin.

19) Put into auxiliary memory - This instruction is identical to the fetch from auxiliary memory command except that a write-in pulse is supplied to the memory instead of a readout pulse.

20) Fetch add (AR1 + addressed word) - This instruction provides for the capacity to add an 8-bit word from the addressed IDB location directly to the contents of the accumulator register AR1. This is useful when calculating average values of sampled data from the IDB.

21) Put A1 into AR2 - This instruction puts the contents of the A1 portion of the accumulator register into auxiliary register AR2. This is used in min-max determination, etc.

22) Put A1 into AR3 - This instruction is identical to the put A1 into AR2 instruction except that AR3 is used instead of AR2.

23) Fetch AR2 - This instruction transfers the contents of auxiliary register AR2 into the A1 portion of the accumulator register. This instruction is used in min-max determination, etc.

24) Fetch AR3 - This is identical to the fetch AR2 instruction except that the contents of AR3 are fetched into A1.

25) Test for asynchronous events - This instruction is used by the processor to check to see if a micrometeoroid event or a fast neutron event has occurred. If an asynchronous event is detected, the processor will enter an appropriate subroutine to process the event.

26) Return address instruction - This instruction is used in conjunction with the "modify instruction" command to modify processor instructions. This command loads its address bits (1 through 10) into a flip-flop register for use by the modify instruction command. The address stored by this instruction specifies the return address to the main program from a subroutine.

27) Modify instruction - This command enters the return address bits of instruction 27 back into the processor instruction memory. This instruction is used in conjunction with instruction 27 to allow the use of common subroutines by the seven main programs.

An example of the use of the above two instructions is shown in Figure 4-39.

4.3 OUTPUT BUFFER

Two output buffers are used. This is to avoid time-sharing problems associated between write-in and readout of data. An approach using one output buffer was considered; however, two output buffers were selected to minimize circuit complexity. In this manner, the processor can perform its operations asynchronously and avoid the severe constraint of time-sharing the output buffers. Thus, the processor loads data into one output buffer while the other is being read out for realtime transmission.

Each output buffer has a capacity of 1200 bits. The size of each frame, however, is programmable by the processor. Note that the processor can store more than one frame of data into an output buffer. This may occur at the lower transmission bit rates where the frame size is approximately 256 bits. The core plane may be arranged as shown in Figure 4-40. Each core is addressable by the 5 and 6 flip-flop registers. The output buffer operation is bit serial; scanning begins at the upper left corner of the plane and proceeds to line 1. When the X axis address counter reaches 32, the Y axis address is upcounted by 1. The X axis counter goes to the zero count and scanning proceeds again to the right, etc. A marker signal is generated when readout is completed from either output buffer. This marker signal is used to synchronize CDS operations and also provides for variable size formatting under the control of the processor program.

The following paragraphs describe the logical implementation involving the loading of the two output buffers and the readout of these output buffers in an alternating fashion to provide continuous, uninterrupted transmission. Figure 4-41 shows a simplified logic diagram for this operation. A marker signal M_1 and M_2 is generated in order to indicate the end of the stored data within the output buffer. This marker signal switches operation from one output buffer to the other by the use of flip-flops A and B. The sampling, processing, and transmission timing relationships are shown in Figure 4-42.

Flip-flop A controls which output buffer is to be loaded (write in) and which output buffer is to be read out. Thus, if A is TRUE, new data

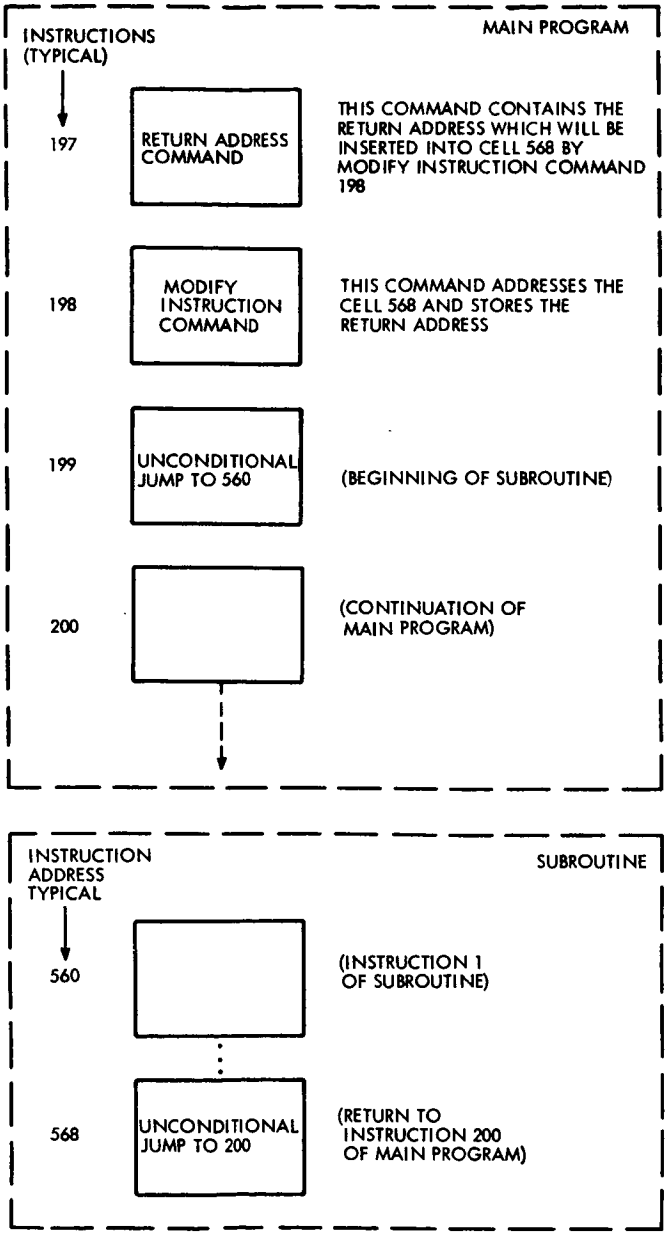


Figure 4-39. Typical Sequence of Subroutine Operation

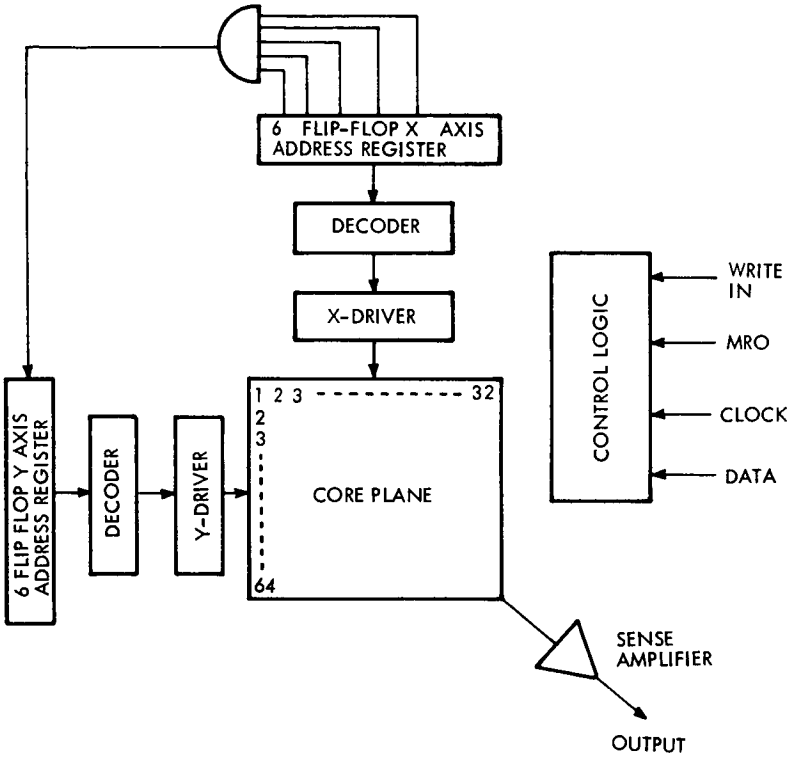


Figure 4-40. Output Buffer Memory

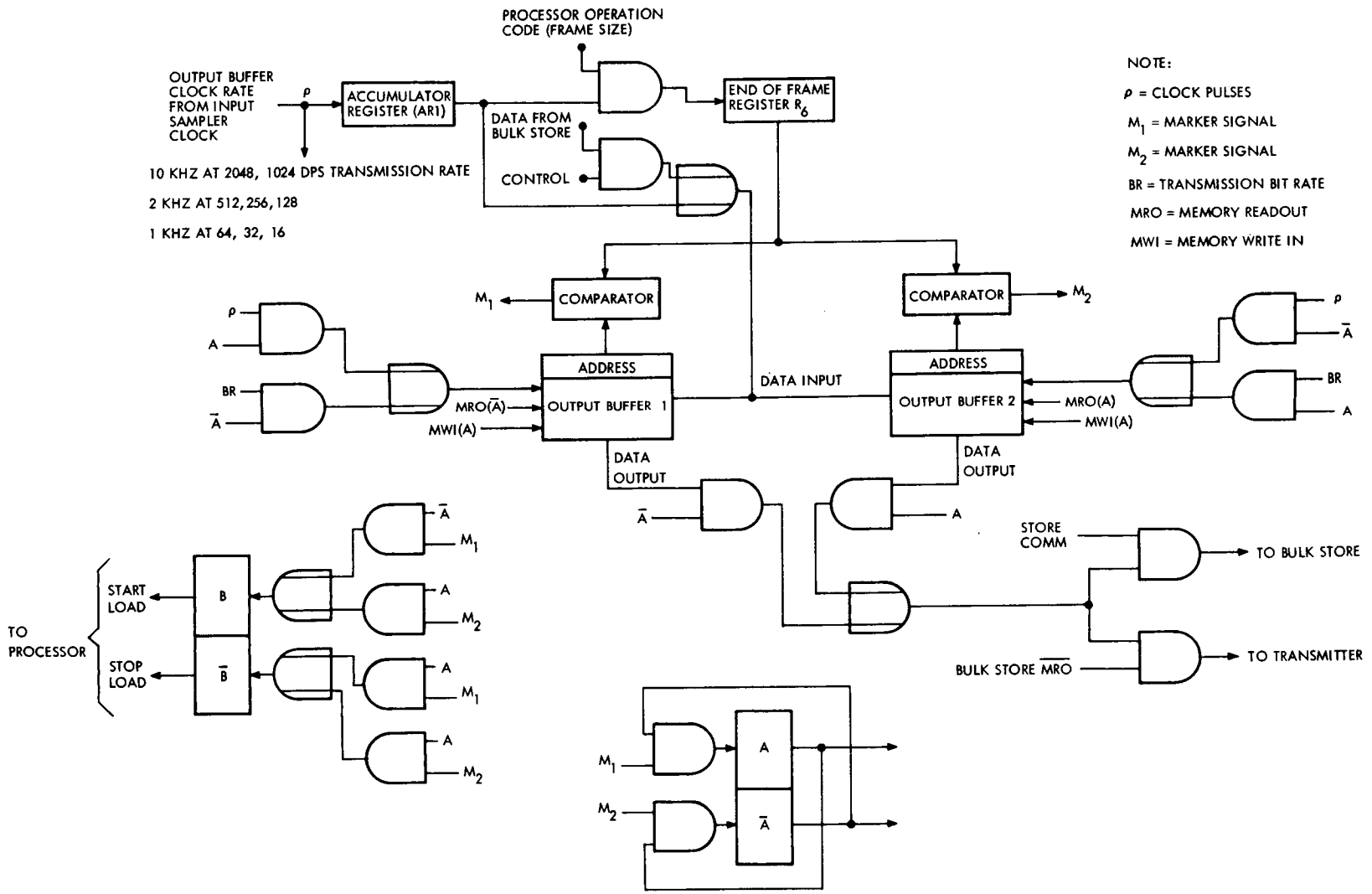
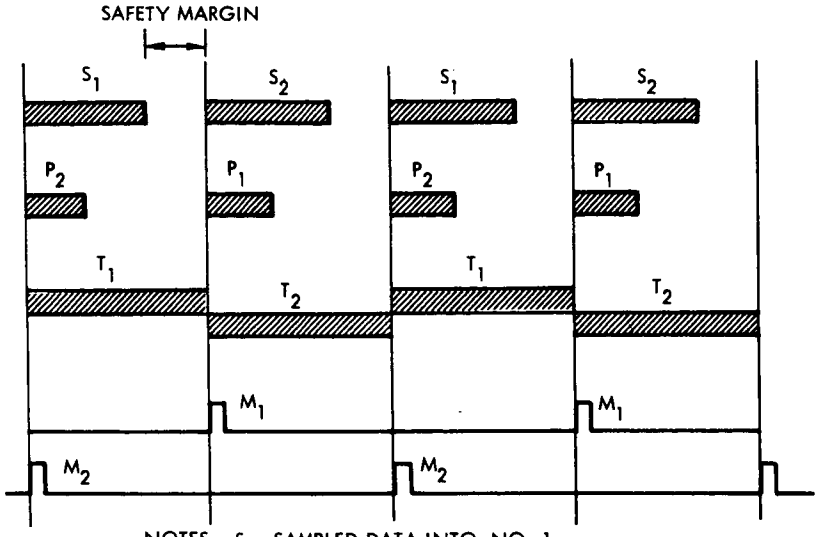


Figure 4-41. Logical Mechanization of Processor-Output Buffer Interface



NOTES: S_1 =SAMPLED DATA INTO NO. 1

S_2 =SAMPLED DATA INTO NO. 2

P_1 =PROCESS NO. 1

P_2 =PROCESS NO. 2

T_1 = TRANSMIT OB NO. 1

T_2 =TRANSMIT OB NO. 2

GROUND RULES:

1) TRANSMISSION TIME> SAMPLING TIME

2) SAMPLING BEGINS WITH OUTPUT BUFFER FLAG (MARKER SIGNAL M_1 OR M_2)

3) PROCESSING BEGINS WITH OUTPUT BUFFER FLAG (MARKER SIGNAL M_1 OR M_2)

4) READ-OUT OF OUTPUT BUFFER IS CONTROLLED BY CRYSTAL CONTROLLED CLOCK (SELECTED BIT RATE)

Figure 4-42. Sampling, Processing, and Transmission Timing Relationship

is loaded into output buffer 1 while data which was previously stored in output buffer 2 is being read out for transmission. Conversely, if A is FALSE, new data is routed to output buffer 2 for storage while output buffer 1 is read out for transmission.

Flip-flop B informs the processor when to start and stop loading data into the output buffers. This controlling element is necessary because the readout rate (which is the bit rate) is slower than the write in rate. That is:

$$WI_{(time)} < MRO_{(time)}$$

As indicated in Figure 4-41, the following rates were selected as the write in clock rate for the five different bit rates:

Write In Clock Rate	Transmission Bit Rate
2 kHz	2048, 1024 bps
1 kHz	512, 256, 128 bps
0.5 kHz	64, 32, 16 bps

By providing a higher write in rate than readout, the processor is given adequate time for processing of data. For example, at 1024 bps 1 second is required to read out one output buffer. By supplying a 10 kbps write in clock:

$$\frac{1024 \text{ bits/memory}}{10 \text{ kbps}} = 0.1 \text{ second}$$

Approximately 0.1 second is required for filling the output buffer and the remaining 0.9 second can be utilized for processing.

The end of frame register R6 stores the address of the last data bit of the transmission frame. This determines the size of transmission frame to be stored and generates marker signals used for CDS timing; i.e., initiates input data sampling cycle, provides switching of output buffers, etc. A detailed CDS block diagram is shown in Appendix D.

4.4 COMMAND PROCESSOR

4.4.1 Requirements

The command processor receives transmission commands from the ground tracking stations to control various spacecraft operations. The command processor performs the following:

1) Accepts a serial data train from ground station up to a frequency of 1 kHz.

2) Accepts, verifies, and decodes transmitted discrete commands:

   - 35 for the CDS
   - 27 for the experiment sensors
   - 22 for other spacecraft subsystems.

3) Reprograms processor instruction via the spacecraft receiver, verifying each instruction prior to entering it into the processor memory.

The CDS commands are listed in Table 4-4.

4.4.2 Command Processor Operational Constraints

The philosophy adopted for command verification prior to execution for the CDS is to provide multiple checks (i.e., parity, complement, and duplication) internally to the spacecraft in lieu of transmitting each command to the ground station for verification. This philosophy has been adopted due to the extensive time required for communication between DSIF stations and spacecraft. For example, at a distance of 2 AU approximately 32 minutes is required to transmit a command to the spacecraft and receive a verification signal back.

The command word length is based upon the required processor instruction memory address word length and its complement plus two word ID bits and one parity bit. Therefore, the command word length is comprised of 23 bits. Figure 4-43 illustrates the command word format. As shown, 2 bits are used to identify whether the following 20 bits are to be used as a processor instruction, processor instruction address, or discrete command codes.

Table 4-4. CDS Command List

1)	Transmission bit rate command 2048 bps
2)	Transmission bit rate command 1024 bps
3)	Transmission bit rate command 512 bps
4)	Transmission bit rate command 256 bps
5)	Transmission bit rate command 128 bps
6)	Transmission bit rate command 64 bps
7)	Transmission bit rate command 32 bps
8)	Transmission bit rate command 16 bps
9)	Realtime data interleaved with stored data mode
10)	Realtime data only mode
11)	Data storage into bulk store mode
12)	High rate engineering data only mode
13)	Self test mode
14)	Enter program verification mode
15)	Fixed realtime mode
16)	Stored data transmission only mode
17)	Switch to redundant A/D converter
18)	Switch to redundant processor instruction memory
19)	Experiment No. 1 power ON-OFF
20)	Experiment No. 2 power ON-OFF
21)	Experiment No. 3 power ON-OFF
22)	Experiment No. 4 power ON-OFF
23)	Experiment No. 5 power ON-OFF

Table 4-4 (Continued)

24)	Experiment No. 6 power ON-OFF
25)	Experiment No. 7 power ON-OFF
26)	Experiment No. 8 power ON-OFF
27)	Realtime data interleaved with stored data for 2048 and 1024 bps transmission rate
28)	Realtime data interleaved with stored data mode for 512, 256, and 128 bps transmission rate
29)	Realtime data interleaved with stored data mode for 64, 32, and 16 bps transmission rate
30)	Transmission of realtime data only mode at 2048 and 1024 bit rates
31)	Transmission of realtime data only mode at 512, 256, and 128 bit rates
32)	Transmission of realtime data only mode at 64, 32, and 16 bps
33)	Data storage operation mode for 2048 and 1024 bps rates
34)	Data storage operation mode for 512, 256, and 128 bps rates
35)	Data storage operation mode for 64, 32, and 16 bps rates

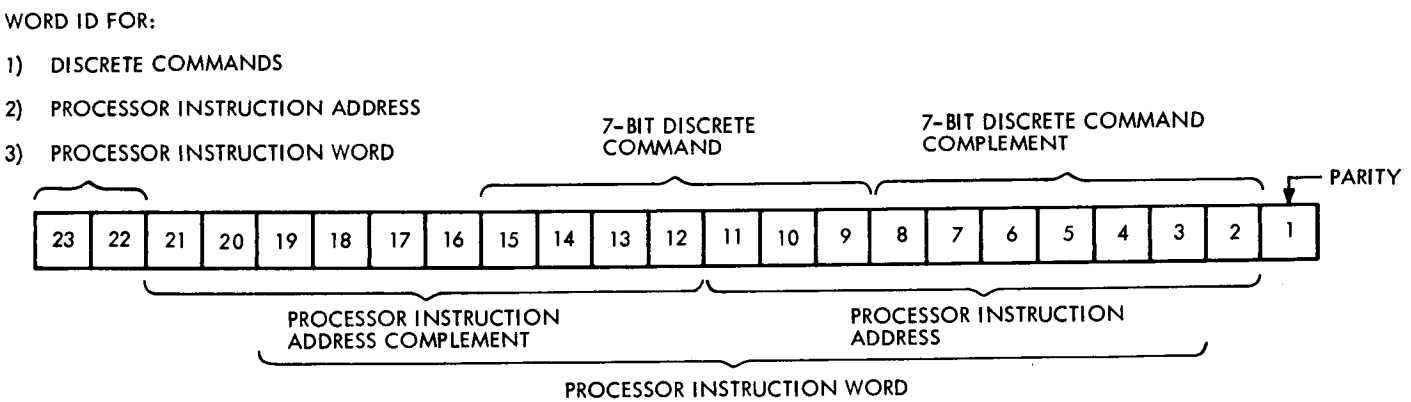


Figure 4-43. Command Word Format

Prior to execution of each command or processor instruction, the command decoder word is checked for proper length and parity. In the case of discrete command and processor instruction words, the received words and their complements are checked for parity and complement prior to execution. In the case of a processor instruction word, the instruction is transmitted twice and verified for both parity and duplication.

4.4.3 Command Processor Design

Figure 4-44 illustrates the block diagram of the command processor. The 23-bit command is serially shifted into the command register by the spacecraft receiver. The command word is verified for length, parity, complement, and duplication as indicated above by the command verifier prior to execution. The word ID selects whether the 20-bit command is transferred to the processor instruction memory or whether the seven discrete command bits are to be decoded by the command decoding matrix into one of 128 discrete outputs.

4.5 BULK STORE

The bulk store is used to store processed data during periods of noncommunication with the DSIF ground stations. Data is processed, formatted, and stored within the bulk store at the prevailing transmission bit rate. Based upon the availability of the DSIF stations, equipped with 85-foot antennas, communication may be established for as little as 11 hours a day. Therefore, 13 hours of storage time are provided by the bulk store.

When communications are re-established, stored data is interleaved with realtime data, alternating on a frame basis. Since the communication period is a maximum of 11 hours per day, 11 hours of real time data will be transmitted with as much as 13 hours of stored data. The data stored over the 13 hour period is reduced by the processor so that it can be transmitted over a period of 5 1/2 hours interleaved with 5 1/2 hours of real time data. The maximum write in bit rate for the bulk store will be 512 bps. The useable storage capacity may be calculated as follows:

$$11 \text{ hours} \times 512 \text{ bps} = 2 \times 10^7 \text{ bits}$$

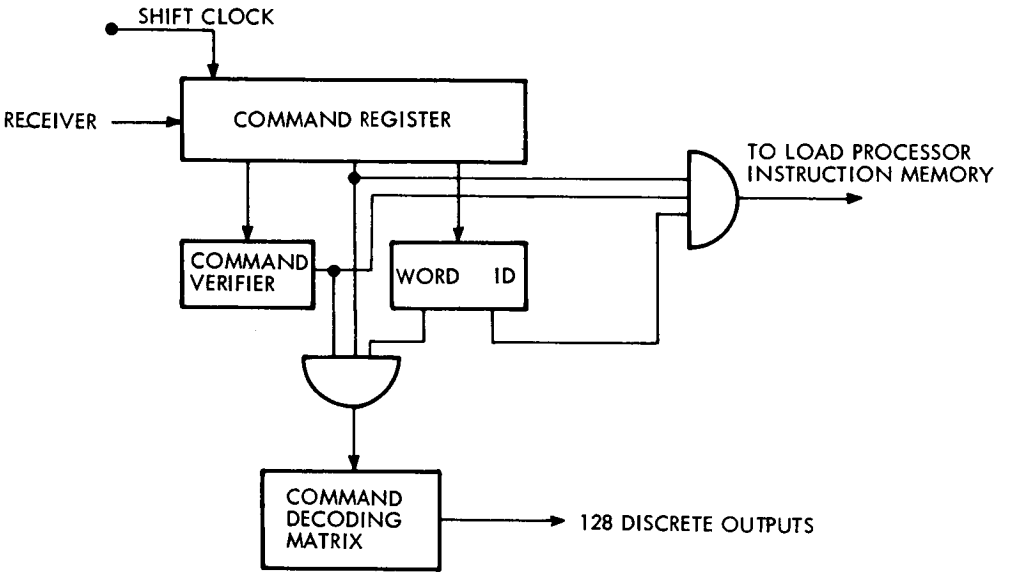


Figure 4-44. Command Processor Block Diagram

Therefore, 2×10^7 bits of tape capacity are required. Based on the present state of the art, a recording density of 1000 bits per inch is available for the bulk store system. The amount of tape required for a 13-hour period is:

$$\frac{2 \times 10^7 \text{ bits}}{10^3 \text{ bits/inch} \times 12 \text{ in/ft}} \cong 1660 \text{ ft}$$

This is the amount of tape required to store the required data. However, this amount must be multiplied by a factor of 2.6 since timing gaps are required between stored data blocks on the tape (512 bits/block). These gaps are required to provide proper timing for the interleaving of bulk store data with realtime data. The actual required amount of tape is 4316 feet.

It is felt that the weight of the extra tape required is insignificant with respect to the advantages of simplicity of output buffer and tape unit design. For example, an output buffer with the capability of simultaneously writing in and reading out was considered, but the complexity of such a system necessitated ruling it out. An incremental tape source (stop start) was also considered but was thought to be impractical for use in the CDS design. The weight of the additional tape (less than 16 ounces) is not a severe penalty to pay in terms of simplicity of design and increased system reliability.

Interleaving is achieved by allowing the processor to load one frame of processed realtime data into an output buffer followed by one frame of stored data from the bulk store within a period of 1 second. That is, during the interleaving mode of transmission, each output buffer is loaded with one frame of realtime data and one frame of bulk store data. The operation is illustrated in Figure 4-45. Figure 4-46 illustrates the timing relationship between processing and transmitting of data when in this mode of operation. At the highest transmission bit rate, the realtime data frame and the bulk store data frame consist of approximately 512 bits each. This means that 1025 bits of data are transmitted from each output buffer each second. When convolutional coding is used, the resultant transmission bit rate is 2048 bps. This assumes the use of one coded bit for each data bit as explained in Section 4.6.

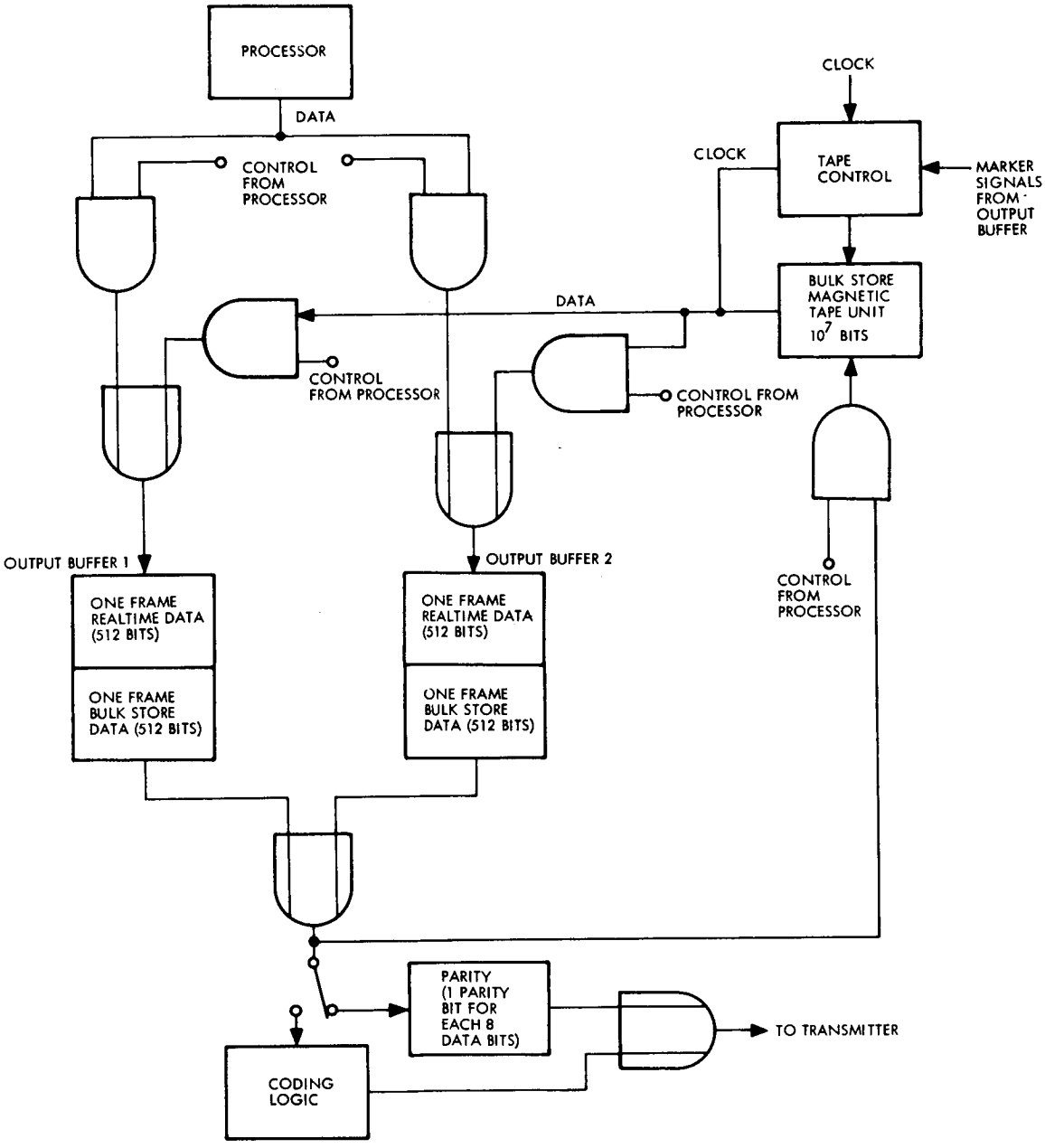


Figure 4-45. Bulk Store Operation Simplified Block Diagram

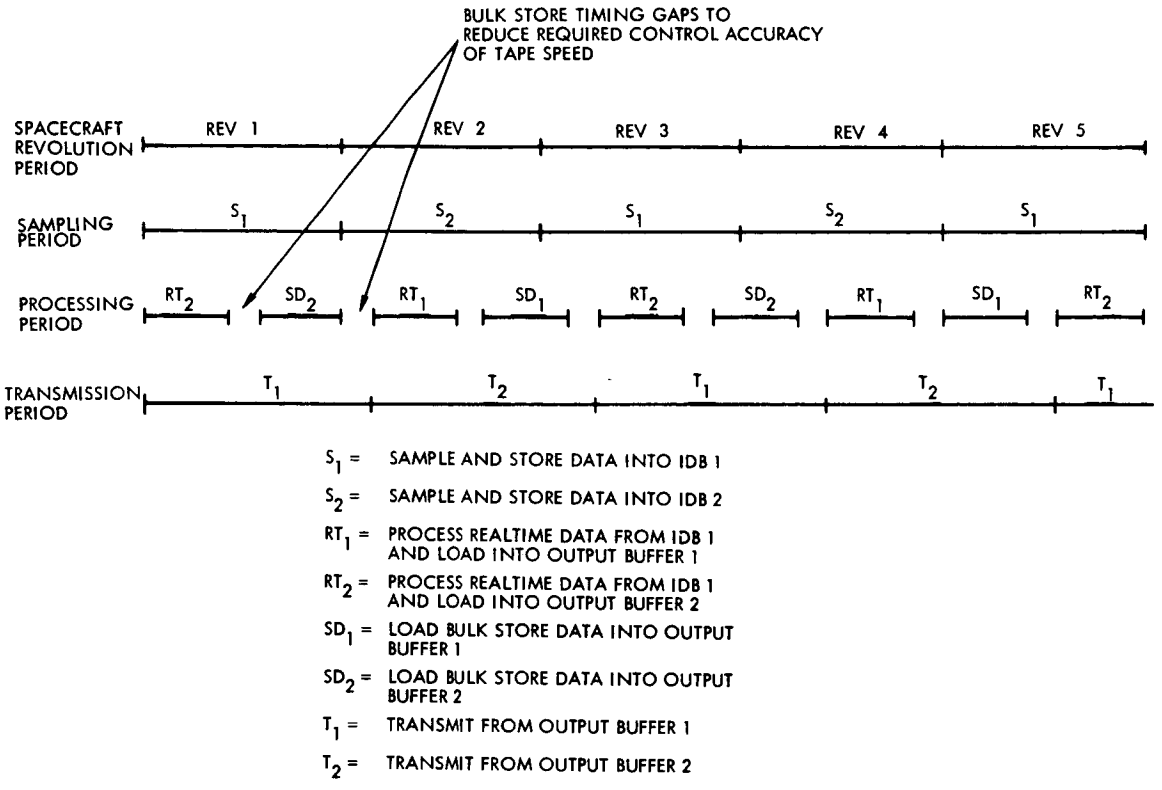


Figure 4-46. Processor Timing for Transmission of Realtime Data Interleaved with Bulk Store Data

Timing gaps are provided (Figure 4-46) to reduce the accuracy requirements of the tape speed control loop. The tape speed is synchronized to the frame rate of the output buffers by use of a proportional control servo loop.

Tape storage systems have been successfully used on spacecraft such as OGO. The reliability estimate for the OGO tape system is estimated to be 0.528 (for a 1 year period); however, more reliable systems are expected to become available for deep space probes as more experience is obtained from the units now flying.

4.6 CONVOLUTIONAL CODING

Convolutional coding was considered for use in the CDS system. The convolutional coding scheme presented here was suggested by NASA/ARC as stated in the introduction. The data is convolutional coded for transmission from the output buffers as data is being read out for transmission.

A simplified block diagram of the coding scheme used is presented in Figure 4-47. As shown, the encoder consists of a 25-bit register, a combiner, and a modulo-2 adder. As bits are shifted into the 25-bit register, the tapes are scanned to determine whether an even or odd number of "1's" are present by use of the modulo-2 adder. If the tapes contain an odd number of "1's," a "1" parity bit is added to the data stream; if the tapes contain an even number, a "0" parity bit is added to the data stream. Therefore, the transmitted bits contain one parity for each data bit.

A gain of 5.6 db is achievable with the use of this coding scheme. As shown in Figure 2-1, convolutional coding is most useful when earth to spacecraft distance exceeds 0.25 AU. Simple parity will be used every 8th data bit when the distance is less than 0.25 AU.

A second and more elaborate convolutional coding scheme was also suggested by NASA/ARC for possible inclusion into the CDS design. This method requires the use of 64 36-bit registers, seven 6-bit registers, one 10-bit register, modulo-2 adders, and control logic. This scheme is not presented within this report because time was not available to adequately analyze its feasibility for use within the CDS.

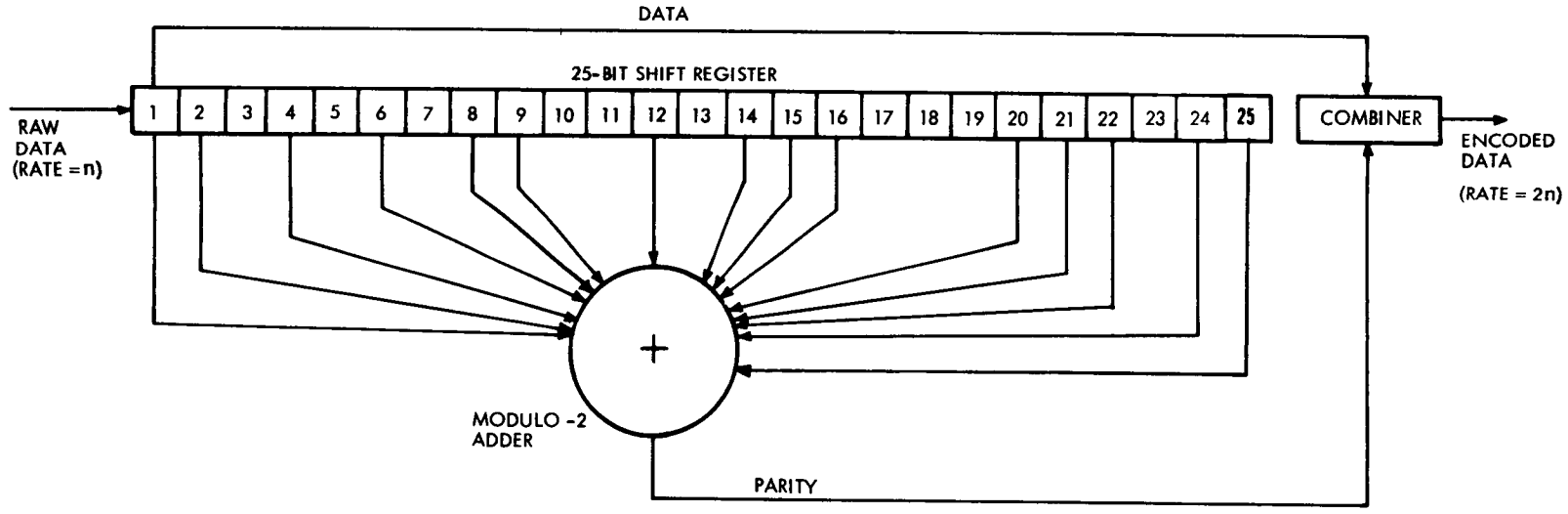


Figure 4-47. Convolutional Coding Diagram

4.7 FIXED REALTIME TRANSMISSION MODE

A fixed realtime mode of operation (Figure 4-48) is provided which directly formats raw data for transmission and bypasses the processor and output buffer. A fixed programmer selects certain data from the IDB in a fixed sequence. For example, data from every 4th or every 8th sector may be selected for transmission. The selected data is then formatted by a combiner which gates frame synchronization and the required identification bits such as format ID, bit rate ID, etc., into the format at the appropriate times. The output from the combiner is fed into the convolutional coding circuitry or simple parity which, in turn, modulates the transmitter modulator. This mode of CDS operation provides for the transmission of a minimal quantity of experiment and engineering data during periods when the processor is being reprogrammed and for reading out the contents of the processor program memory (program verification mode).

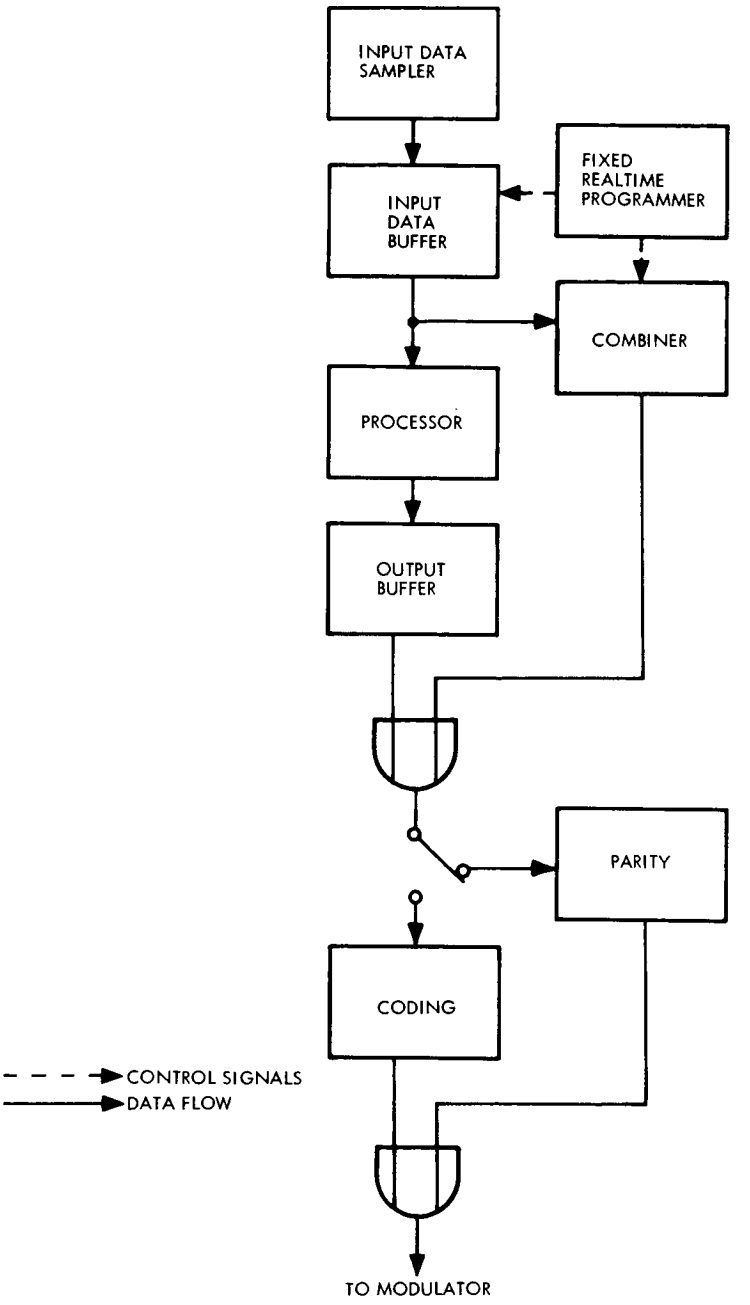


Figure 4-48. Fixed Realtime Mode Simplified Block Diagram

5. WEIGHT AND POWER SUMMARY

The weight and power assessment for the CDS is based upon the use of low power integrated circuit logic such as the Fairchild 9000 series. The Fairchild 9000 series integrated circuits were selected due to their low power consumption and their off-the-shelf availability.

Weight and power calculations for the CDS memories are based on the use of core systems. The following ground rules are used in calculating CDS memory average power:

1) Core drive currents will be 300 ma with an applied voltage of 20 volts.

2) 1 μsec read-write pulse time is used.

3) 2 μsec inhibit pulse time is used.

4) Sense amplifier power is pulsed on for 5 μsec during read cycles.

5) Operating frequency

   - IDB - Write in - 2.3K memory cycles/sec
     Readout - 4.6K memory cycles/sec
     } each memory
   - Processor instruction memory - 1.2K memory cycles/sec (max)
   - Auxiliary memory - 100 memory cycles/sec (average)
   - Output buffer - Write in - 2.4K memory cycles/sec (max) - each memory
     Readout - 1K memory cycles/sec (max) - each memory.

IDB operating frequency is based upon the use of 36 x 64 eight-bit words. That is, there are 64 sector cells and 36 words for each sector cell which equals 2304 words. These 2304 words are stored into the IDB in one revolution or one second. Therefore, the operating frequency for write in equals approximately 2.3K memory cycles per second. Assuming these 2304 samples must be sampled and processed within 0.5 second, the readout frequency is twice read in or 4.6K memory cycles per second.

The processor memory operating frequency assumes the use of 300 instructions per program. Assuming that processing must be

complete in 0.5 second, 600 memory cycles per second are required. Since each instruction must be restored, 600 read and 600 restore cycles occur per second requiring an operating frequency of 1.2K memory cycles per second.

Power is turned off on the redundant memory when not in use. The auxiliary memory is used primarily for storing fast neutron experiment data. The write in rate is approximately 20.7K bits in one hour each 24 hours. Assuming a 10-bit word, a write in cycle time of less than one Hz is required. For readout, 2070 10-bit words are to be read out over a period of 20 hours. Therefore, the average cycle time is less than one memory cycle per second.

$$\frac{2070 \text{ words hr}}{20 \text{ hrs } 3600 \text{ sec}} < < 1 \text{ Hz}$$

The auxiliary memory is also used by the processor to perform special operations such as calculating average values of data over many spacecraft revolutions. For this auxiliary memory, an overall average cycle time of 100 memory cycles per second is assumed.

The output buffers require a write in rate of 1200 bits in 0.5 second or 2.4 kHz. The maximum rate for readout is 1024 bits per second based on the maximum transmission bit rate of 2048 (1 bit parity is added for each data bit since parity bits were not entered into the output buffer).

Table 5-1 summarizes the calculated weight and power for the CDS.

Table 5-1. CDS Weight and Power Summary

Subsystems	Power (watts)	Weight (pounds)
Input data sampler logic	2.2	1.5
Processor logic	2.2	1.5
Command processor*	low average power - unit turned off when not used	0.5
Power converter	--	3.0
Memories		
1) Two input data buffers (12K bits each)	0.43	2.5
2) Two output buffers (1.2K bits each)	0.16	2.0
3) One auxiliary memory (30K bits)	0.10	2.5
4) Two processor memories (18K bits each)	0.26 for one	3.0
Total	5.35 (8.9 watts raw power)**	16.5

*Peak power = 1.7 w.

**Total Power In, assuming 60% converter efficiency is:
$P_{in} = 5.35/0.6 \cong 8.9$ watts.

6. CDS RELIABILITY ASSESSMENT

The CDS reliability is calculated based upon component failure rates suggested by NASA/ARC; these component failure rates are tabulated in Table 6-1. The rates listed are similar to those used to calculate the reliability for the Pioneer VI data handling subsystem. This allows a comparative analysis between the CDS and the Pioneer VI system reliability using the same ground rules.

Table 6-1. Component Failure Rates and Usage

Part Type	Failure Rates Per 10^9 Hours	Usage
Resistor, carbon composition	10	900
Diode, silicon, switch	18	2176
Transistor, silicon, switch	30	440
Transformers, switching	10/ winding	490
Memory cores	0.01	93, 104
Integrated circuits	10	1748

CDS reliability is calculated for three mission times: 1 year, 1.5 years, and 2 years. The mission phase reliabilities for these three time periods are listed in Table 6-2 for each of the CDS reliability blocks. Table 6-3 lists the component usage for each block.

The following equation is used to calculate all CDS reliability numbers:

$$P_s = e^{-\lambda t K}$$

where

P_s = probability of success
λ = failure per 10^9 hours
t = mission time

A constant $K = 1$ was assumed. This constant specifies environmental and packaging factors.

Table 6-2. CDS Reliability Calculated for Three Time Periods

No	Description	Failures/10^9 Hours	Mission Phase Reliabilities		
			1 Year	1.5 Years	2 Years
1	Clock	525	0.995	0.993	0.991
2	Fixed programmer	420	0.996	0.994	0.993
3	Pulse gates	300	0.997	0.996	0.995
4	Analog gates (A)	240	0.998	0.997	0.996
5	Analog gates (B)	360	0.997	0.995	0.994
6	Bi-level gates	375	0.997	0.995	0.993
7	Fast neutron gates	150	0.999	0.998	0.997
8	Main multiplexer	825	0.993	0.989	0.986
9	Sub-multiplexer	300	0.998	0.996	0.995
10	Counters	600	0.995	0.992	0.990
11	A/D converter	287	0.999	0.999	0.999
12	Experiment control	150	0.999	0.998	0.997
13	Sync signals	300	0.997	0.996	0.995
14	Input data buffer logic	225	0.998	0.997	0.996
15	Input data buffer	12859	0.989	0.976	0.959
16	Accumulator register	1125	0.990	0.985	0.980
17	Control logic	1500	0.987	0.980	0.974
18	N1 + N2 registers	750	0.993	0.990	0.987
19	Adder/subtractor	375	0.997	0.995	0.993
20	Log scaling logic	375	0.997	0.995	0.993
21	Formatter	150	0.999	0.998	0.997
22	Pulse height analyzer	225	0.998	0.997	0.996
23	Counter registers (A)	750	0.993	0.990	0.987
24	Counter registers (B)	375	0.997	0.995	0.993
25	Fixed realtime logic	375	0.997	0.995	0.993
26	Auxiliary memory	22428	0.822	0.745	0.675
27	Processor memory	21237	0.971	0.941	0.903
28	Output buffer logic	210	0.998	0.997	0.996
29	Output buffer	2758	0.953	0.930	0.907
30	Bulk storage	101447	0.411	0.264	0.169
31	DC power decouplers	660	0.994	0.991	0.989
32	Block coding	750	0.993	0.990	0.987
33	Biphase modulator	60	0.999	0.999	0.999
34	Command processor	1500	0.987	0.980	0.974

Table 6-3. CDS Component Summary

	Description	Components Used
1.	Clock	35 integrated circuits (IC)
2.	Fixed programmer	28 IC
3.	Pulse gates	20 IC
4.	Analog gates (A)	16 IC
5.	Analog gates (B)	24 IC
6.	Bi-level gates	25 IC
7.	Fast neutron gates	10 IC
8.	Main multiplexer	55 IC
9.	Sub-multiplexer	20 IC
10.	Counters	40 IC
11.	A/D converter (2 each)	12 IC 2 transistors 4 resistors 1 zener
12.	Experiment control	10 IC
13.	Sync signals	20 IC
14.	Input data buffer logic	15 IC
15.	Input data buffer (2 each)	100 pulse transformers (3 windings each) 256 diodes 100 resistors (carbon) 12,000 cores 100 IC 50 transistors 800 stack connections
16.	Accumulator register	75 IC
17.	Control logic	100 IC
18.	AR2 and AR3 registers	50 IC
19.	Adder/subtractor	25 IC
20.	Log scaling logic	25 IC
21.	Formatter	10 IC
22.	Pulse height analyzer	15 IC
23.	Counter registers	50 IC
24.	Counter registers	25 IC
25.	Fixed realtime logic	25 IC

Table 6-3 (Continued)

	Description	Components Used
26.	Auxiliary memory	100 pulse transformers (3 windings each) 512 diodes 100 transistors 200 resistors (carbon) 30,000 cores 200 IC 1500 stack connections
27.	Processor memory (2 each)	75 pulse transformers (3 windings each) 512 diodes 200 resistors (carbon) 18,352 cores 200 IC 1000 stack connections
28.	Output buffer logic	14 IC
29.	Output buffer (2 each)	20 pulse transformers (3 windings each) 64 diodes 50 resistors (carbon) 1200 cores 20 transistors 20 IC 100 stack connections
30.	Bulk store	Part breakdown is not presented for this unit.
31.	DC power decoupler	25 resistors (carbon) 35 capacitors (ceramic)
32.	Convolutional coding	32 IC
33.	Biphase modulator	4 IC
34.	Command processor	100 IC

6.1 FIXED REALTIME MODE RELIABILITY

CDS reliability was calculated for various missions. Figure 6-1 shows the reliability diagram for the fixed realtime transmission mode. The blocks numbered 15 specify the two input data buffers. The two memories are shown as 15a and 15b; both memories are required for the fixed realtime mode. The A/D converters 119 and 116 are considered redundant as shown in the figure. The reliability numbers calculated for this mode are as follows:

1 year mission - 0.761
1.5 year mission - 0.689
2 year mission - 0.626

Fast neutron experiment blocks 7, 22, 23, and 24 are not entered into the reliability number calculated above, since the processor is required to sample and process fast neutron data. The processor is bypassed in this mode.

6.2 PROCESSOR RELIABILITY

Figure 6-2 shows the reliability diagram for the processor portion of the CDS. The reliability numbers shown below are based upon the use of a redundant processor memory as indicated by 27a and 27b in Figure 6-2.

1 year mission - 0.935
1.5 year mission - 0.890
2 year mission - 0.838

6.3 OVERALL CDS RELIABILITY

Figure 6-3 shows the reliability of the entire CDS. Blocks 3, 4, 5, 6, and 7 are given a relative importance to mission success by the percentage number shown. For example, block 4, which represents the analog gates, is given a relative importance of 45 percent to mission success. An assumption was made that the functioning of any combination of blocks which add up to 80 percent would constitute a mission success. Blocks 18, 19, and 20 are treated similarly. In this calculation, blocks 11a and 11b, A/D converters, are considered redundant and blocks 27a and 27b, processor instruction memory, are also considered redundant.

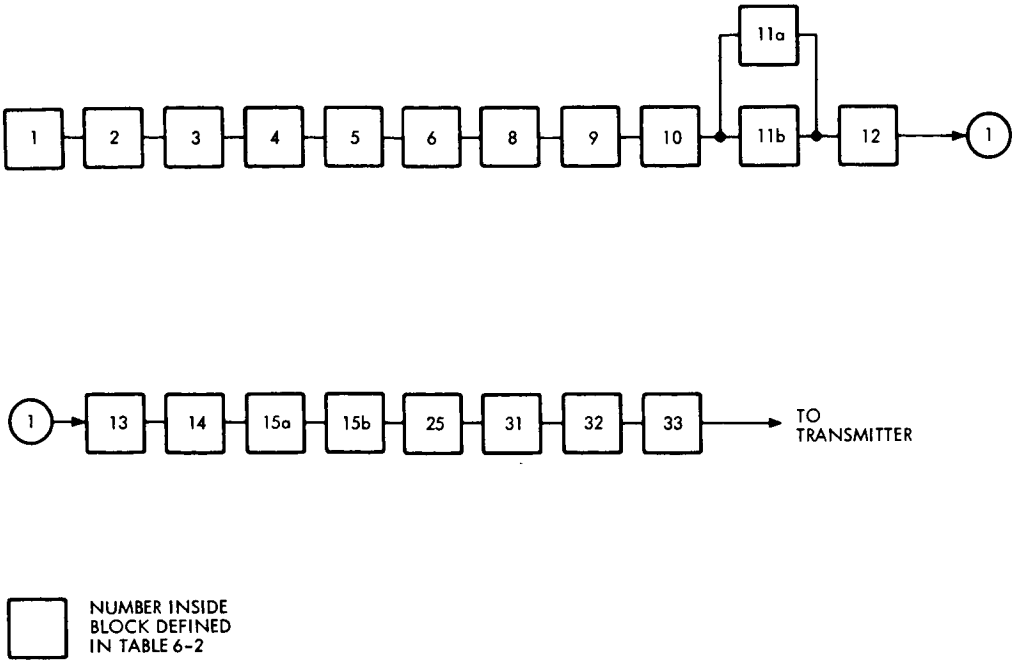


Figure 6-1. Reliability Block Diagram of Fixed Realtime Transmission Mode

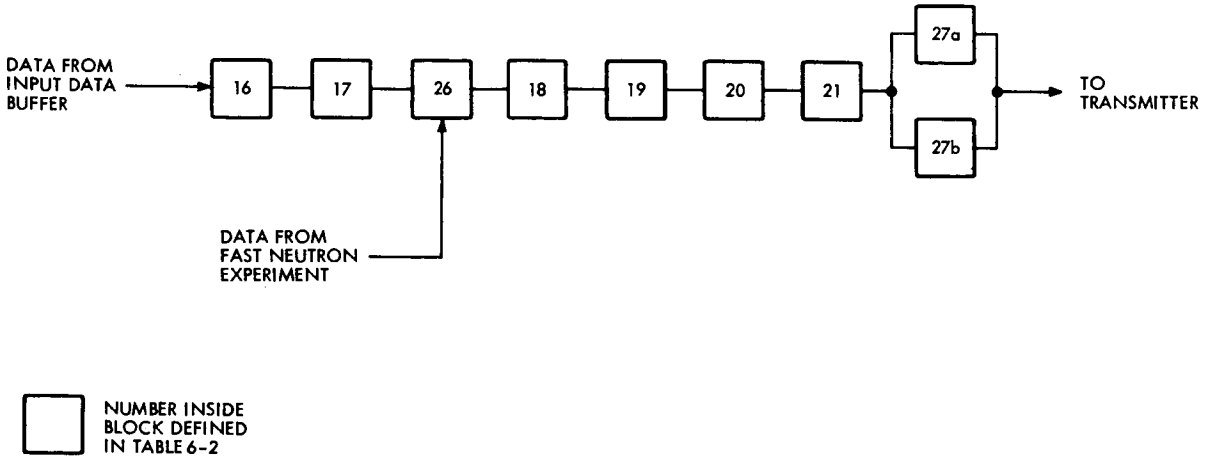


Figure 6-2. Processor Reliability Block Diagram

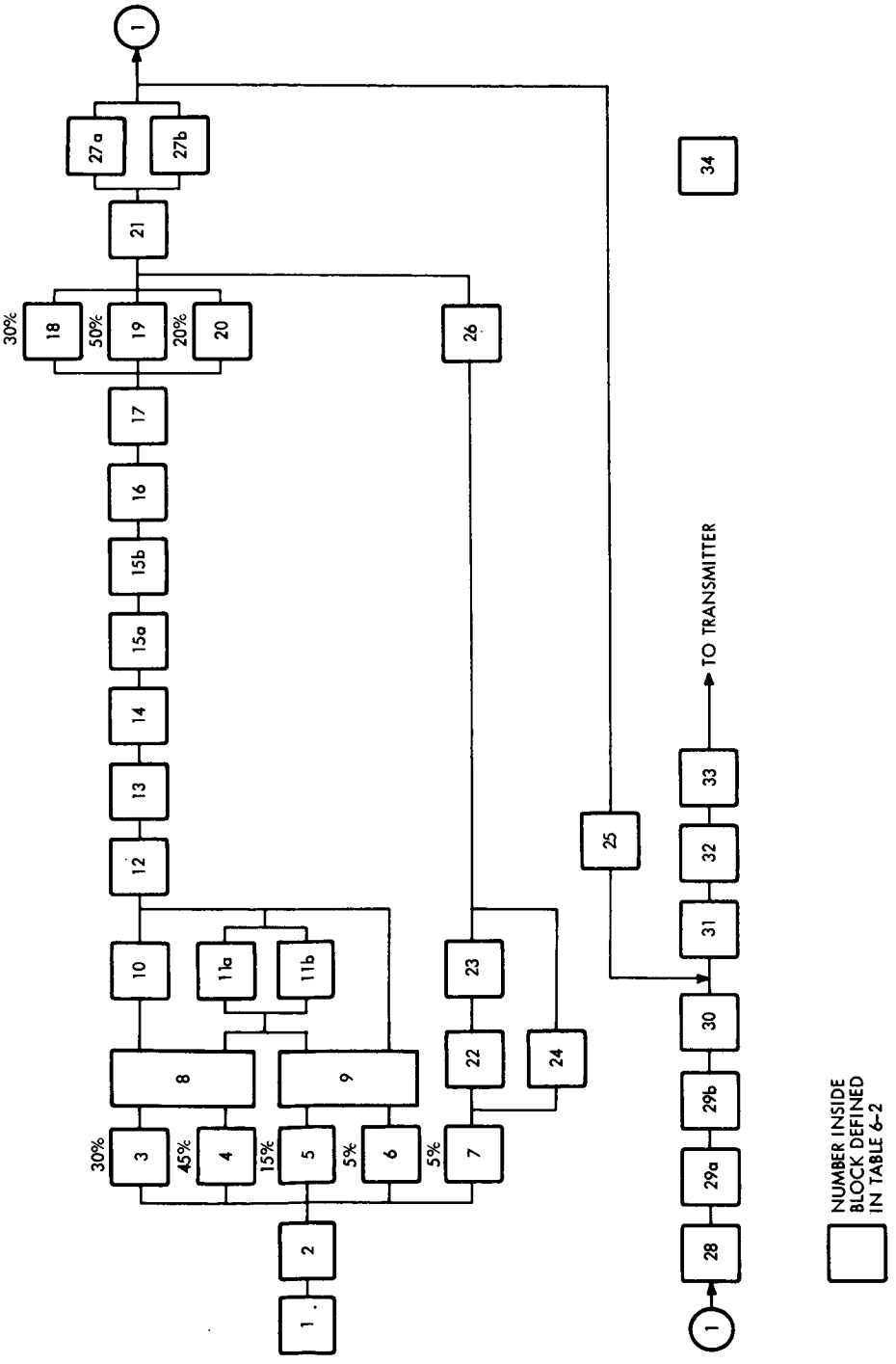


Figure 6-3. Reliability Block Diagram

Both IDB's 15a and 15b and both output buffers 29a and 29b are required as shown in the figure. The reliability numbers calculated for the CDS based upon the above conditions are as follows:

1 year mission - 0.277
1.5 year mission - 0.143
2 year mission - 0.073

The low reliability number results from the use of a magnetic tape unit which has the following reliability number:

1 year mission - 0.411
1.5 year mission - 0.264
2 year mission - 0.169

These numbers are based upon the tape unit used in the Orbiting Geophysical Observatory (OGO). If the tape unit reliability is removed from the basic calculations, the CDS reliability numbers for Figure 6-3 are:

1 year mission - 0.673
1.5 year mission - 0.546
2 year mission - 0.434

A reliability number was also calculated with all the reliability blocks of Figure 6-3 in line. The reliability figures calculated are as follows:

1 year mission - 0.277
1.5 year mission - 0.143
2 year mission - 0.073

Excluding the tape unit, block 30, the following line reliability numbers result:

1 year mission - 0.672
1.5 year mission - 0.542
2 year mission - 0.433

A failure rate is assigned to CDS connections; for example, ten connections are assumed for all integrated circuits. The failure rate assumed for connection is 0.1×10^{-9} failures per hour.

A summary of the overall CDS reliability is presented in Table 6-4.

Table 6-4. Summary of CDS Reliability

Reliability Number	1 Year Mission	1.5 Year Mission	2 Year Mission
Fixed realtime transmission mode	0.761	0.689	0.626
CDS processor only	0.935	0.890	0.838
Tape recorder only	0.411	0.264	0.169
CDS (80% data retrieval and tape recorder)	*0.277	0.143	0.073
CDS (80% data retrieval no tape recorder)	0.673	0.546	0.434
CDS (all elements in-line with tape recorder)	*0.277	0.143	0.073
CDS (all elements in-line no tape recorder)	0.672	0.542	0.433

*NOTE: The differences between the two numbers are only evident in the fourth decimal place.

7. SYSTEM SIMULATION PHILOSOPHY

A discussion of the simulation philosophy is presented in the following sections. This simulation philosophy is useful for studying the following aspects of the CDS:

1) Test timing relationships. This involves testing the effects of differences in spacecraft rotation, transmission rates, processing rates, etc. While the proposed system timing appears to be relatively straightforward, nevertheless simulation runs with emphasis upon timing may uncover potential conflicts that can establish confidence in the planned timing.

2) Validation of processor programs. Processor programs can be validated as they are written to verify both timing and the ability to process data as expected. The main value of this will be the ability to quickly determine the effect of changes in programs.

3) Malfunction analysis. Certain types of failure modes can be investigated to determine system performance under degraded conditions; for example, partial failure of input or output buffers.

4) Investigation of possible design changes. This may be closely related to 3) above. For example, changes in processor control circuitry can be tested to investigate the ability to program around certain types of failures. Tradeoff studies can be conducted on different command lists for the processor. Variations in buffer sizes may be studied.

5) Format analysis. Present plans provide for short frame lengths when transmission rate is low. Studies can be conducted to find the factors which determine optimum frame size and format.

6) Assess memory size and speed requirements. This will maximize CDS efficiency in terms of required clock rates and required processor instruction memory commands.

7) Simulation and process. Simulate various types of input functions and process simulated inputs using typical processor subroutines to evaluate signal processing capability of the CDS.

The discussion consists of a general description of each simulation routine and a brief description of its operation. A modular approach to the simulation is discussed where separate routines are used to simulate functions such as a data generator, sun pulse generator, input data buffer, processor, processor programs, etc. This modular approach to the simulation enables changes to be incorporated in one routine without affecting others, thus providing a system simulation with a high degree of flexibility.

7.1 GENERAL DESCRIPTION OF SIMULATION PHILOSOPHY, ROUTINES, AND OPERATION

The simulation is made up of a number of routines which interact with each other to simulate the system. The effort has been made to make the routines as independent of each other as possible so that the level of simulation detail can be modified in different parts of the system with a minimum overall effect upon the simulation programming.

Figure 7-1 shows the basic simulation routines. Simscript can be used to write the routines for the simulation program. Because of limitations in the naming of routines, the Simscript names do not correspond exactly to the block names shown in Figure 7-1. Simscript is a language developed by Rand Corp. to facilitate digital simulation of complex systems. There are many other languages, but Simscript has remained as one of the standards by which new simulation languages are evaluated. It provides the capability of quickly simulating complex systems because of a number of unique simulation oriented instructions. These include such commands as create, destroy, cause, cancel, remove, file, find maximum, etc., which are particularly suitable to the needs of simulation.

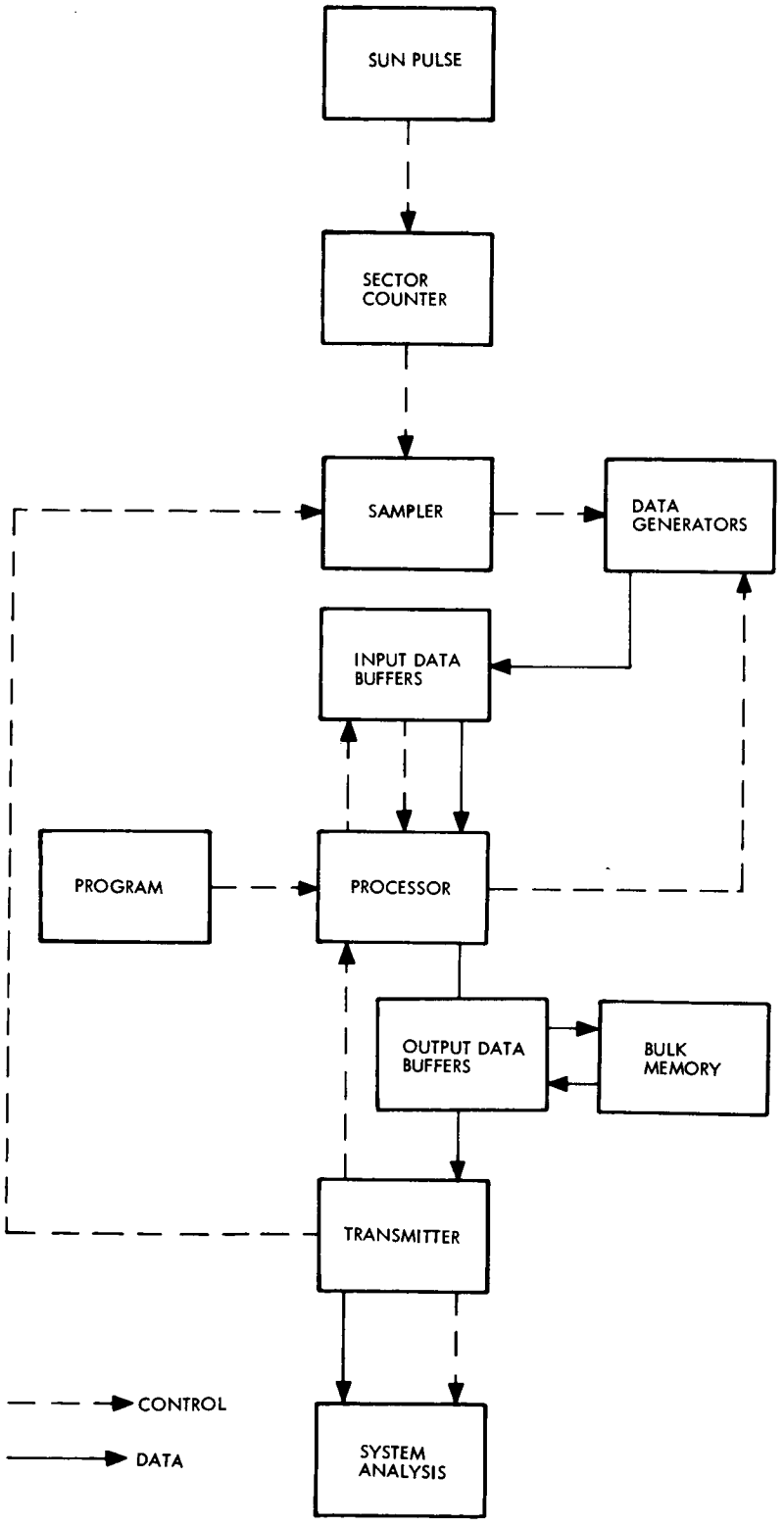


Figure 7-1. Simulation Routine Block Diagram

7.1.1 Sun Pulse

Provision is made to specify the seconds per rotation (spr) and a drift rate (DRIFT). Also, the length of a simulation run is specified in terms of the number of revolutions of the spacecraft, that is, the total number of sun pulses (NUM(SUN)). For example, a run may be specified with the following values:

Seconds per rotation = 1.01 sec

Drift = -0.0001 sec

Number of sun pulses = 200

This would result in the rotation time gradually slowing down from 0.99 seconds per revolution to 1.01 seconds. In Simscript two programs are used. The first, called START, reads in spr, DRIFT, NUM(SUN), transmission rate, processor clock rate, and the program selected. The second routine, called SUN, computes the time at which each sun pulse will occur. Whenever a sun pulse occurs, the sector counter routine (SECTR) is initiated and the sector count started.

7.1.2 Spacecraft Sector Counter

The sector counter routine, called SECTR, simulates in detail Figure 4-4. In particular, the divide-by-64, the B register (OREG), and the sun sector counter register (SCNTR) are simulated. As each sun sector count is generated, the sampler routine (SAMPL) is called to initiate the sampling sequence.

7.1.3 Input Data Sampler

This routine, called SAMPL, is called every sector by SECTR; however, samples will not actually be taken unless the sampler has been enabled by a control pulse from the output data buffers. When this control pulse has been received, then samples will be taken for the next 64 sector counts. SAMPL will call a number of data generator routines which in turn will store data into the proper location of the input data buffers.

7.1.4 Data Generator

The data generators provide the opportunity for true flexibility for the simulation. The data generators are actually a collection of independent subroutines with names such as MAG, VLF, COS, etc. Each data generator creates an 8-word data block called DATA. Stored within DATA will be the identification of the type of data, the type of processing required, either the actual data word or the number of bits, the type of output analysis required, and other special parameters. DATA is stored in the proper position in an input data buffer. There are three possible situations relating to the way in which data is introduced. First, if the processing time for a particular type of data is always constant, then rather than simulate the processing program in detail, we can merely generate a suitable delay to simulate the processing time. In this case, one of the special parameters would be the number of processor clock cycles required for processing. In the second case, we may want to simulate the actual processing of data either to verify the processor program or to accurately reflect processor time when the time for processing is a function of the data value. In this case, we would generate, according to a suitable function, actual data values to flow through the system. The third case provides for inputting real experiment data into the system to observe the relationship between data input and data output. For this case actual data values would be read into the system and properly sampled and processed.

7.1.5 Input Data Buffers

The input data buffers are nothing more than two 64 by 33 storage arrays. They are called IDB1 and IDB2. The address of DATA is stored in the proper location by the appropriate data generator. A possible error condition which is looked for by the simulation is the attempt to store data in an input data buffer before the previous data has been processed by the processor. This is determined by the presence of a nonzero value in a buffer location where data was extracted by the processor. The processor will store a zero in each buffer location where data has been extracted. Conversely, if the processor extracts a zero value from the buffers, this will also be flagged as an error since no data word consists of all zeros.

7.1.6 Processor and Program

Since the program is an intimate part of the processor, these two blocks are discussed together. Certain arbitrary decisions have been made with regard to these elements. It is believed that these decisions add to the flexibility without detracting from the system realism. This program (PROG) contains a list referring to locations in the input data buffers. The processor subroutine (PROCS) refers to this list in a fixed order. The typical procedure for the PROCS subroutine is as follows:

1) PROCS receives signal from output data buffer to start processing data.

2) Set program pointer to correct starting point in PROG list.

3) Obtain IDB location from PROG list.

4) Obtain DATA block address from IDB location.

5) Based upon parameters stored in DATA, simulate the processing of data. Determine the time required for processing. Set all appropriate IDB words to zero. Place address of DATA in ODB.

6) Increment program counter.

7) If processing complete, stop PROCS.

8) After a delay equal to time for processing, return to 3).

Note that the processor refers back to DATA to obtain information on how to process the data. The simulation of data processing is actually part of the data generator routine. This means that when we rewrite a data generator routine, within one routine we can provide not only for the changes in the type of data but also the processing of the data.

7.1.7 Output Buffers

These buffers are called ODB1 and ODB2. They are not subroutines nor are they fixed two-dimensional arrays like IDB1 and IDB2. Rather, they are sets of data words filed in a FIFO (first in -- first out) list. The processor routine generates one or more dummy DATA blocks to

provide for frame sync and other identifiers. In addition, certain counters are used to keep track of the number of bits stored in each ODB.

7.1.8 Transmitter

This is a subroutine called TRANS. TRANS removes DATA blocks from the output data buffers and calls the system analysis routine for output reports. In addition, TRANS also stores data, when required, in bulk memory.

7.1.9 Bulk Memory

Bulk memory, called BULK, is similar to the output data buffers in that it is a variable size set of DATA blocks.

7.1.10 System Analysis

This routine controls a collection of various report writing subroutines. One report provides a chronological listing of the data transmitted, while others accumulate various statistics for summary reports on the system operation.

8. CDS TEST AND CHECKOUT PHILOSOPHY

Three types of testing are recommended to verify performance of the CDS:

- System integration testing
- Pre-launch testing
- In-flight testing.

8.1 SYSTEM INTEGRATION TESTING

The test philosophy adopted is one which automatically tests the system under all possible modes of operation. Figure 8-1 illustrates the system integration test block diagram. The test system consists of an external computer, an experiment simulator and stimulator, and a 28-vdc power supply. An external computer is used because there are many combinations of tests to be performed and the time required for this type of testing is extensive. The experiment simulator, under computer control, provides a known simulated input directly into the CDS, bypassing the experiment sensors. The experiment stimulator supplies actual radiation and magnetic signal sources to directly stimulate the experiment sensors. A variable external +28-vdc power supply is provided to enable test of CDS and sensor performance under varying primary voltage conditions.

Two categories of tests are performed during the integration test. One utilizes the closed loop system in which the external computer controls the experiment simulator and external power supply. The other test uses the experiment stimulator.

A test program is loaded into the processor memory which will process simulated experiment inputs as will actual sensor inputs. The test program verifies the operation of the input data sampler and the processor and its associated circuits. That is, this philosophy provides for detail testing of all elements of the CDS under varying input data rates. In particular, the processor memory is thoroughly tested to insure reliable operation (i.e., no bit dropouts). These tests are performed under power supply voltage sun pulse period variations. Variation in the sun pulse period simulates changes in spacecraft spin

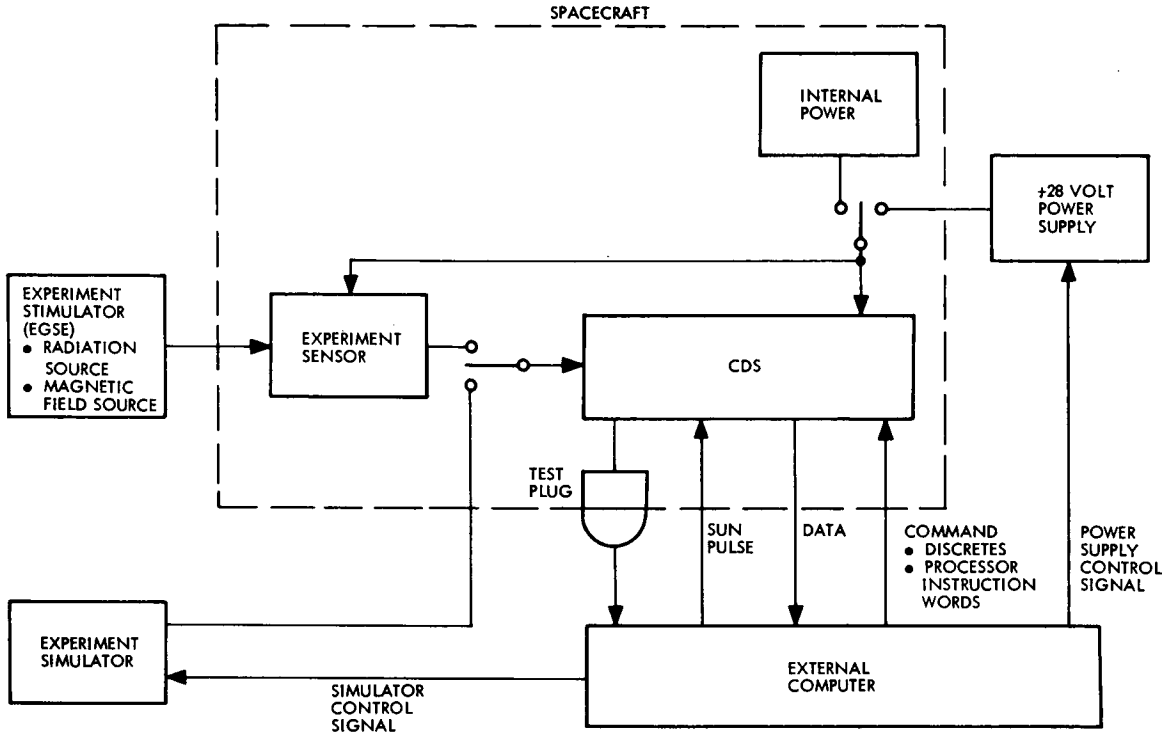


Figure 8-1. System Integration Test Block Diagram

rate. In addition, the external computer performs certain diagnostic routines which is used for fault isolation. Certain diagnostic routines which are determined by previous test results are inserted into the CDS instruction memory.

A test plug is provided for monitoring certain critical signals within the CDS.

8.2 PRE-LAUNCH TESTING

During the pre-launch phase of testing, an external computer will be used to monitor flight programs.

Figure 8-2 illustrates a pre-launch system test configuration. The pre-launch system test consists of the same external computer and experiment stimulator that were used during system integration test. Pre-launch testing will be limited to verification of the performance of the internally programmed and fixed modes of operation. That is, each mode will be commanded and the CDS will be verified for proper operation by the use of the experiment stimulator. The external computer will verify each CDS transmission for proper data processing and formatting.

The CDS self test mode will also be used to verify system operation during pre-launch phases. Flight programs will not be changed during pre-launch testing unless required in order to prevent accidental re-programming of the CDS.

8.3 IN-FLIGHT TESTING

Internal self test routines are used to periodically check CDS operation. These self test routines can be modified by ground command if required.

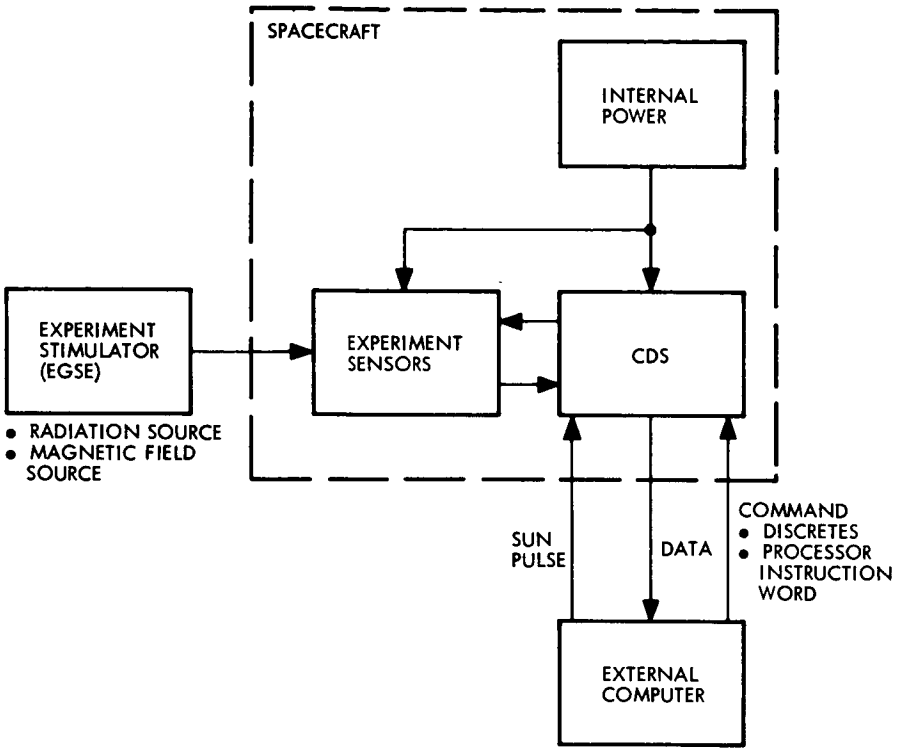


Figure 8-2. Prelaunch System Test Configuration

9. CENTRAL DATA SYSTEM COMPARISON TO PIONEER VI DATA HANDLING SYSTEM

9.1 INTRODUCTION

The purpose of the CDS study is to determine an efficient data format and data system for a lightweight deep space probe. A requirement of the study contract is to compare the CDS design to a baseline system. The Pioneer VI data handling subsystem was chosen as the baseline system by mutual agreement between TRW Systems and NASA/ARC.

9.2 PIONEER VI DATA HANDLING SUBSYSTEM DESCRIPTION*

The Pioneer VI data handling subsystem is made up of a digital telemetry unit (DTU), a data storage unit (DSU), and a command decoder. Redundant command decoders and redundant programmers within the DTU are used in the Pioneer VI system. A system block diagram is shown in Figure 9-1. The six experiment sensors are shown along with their individual processing blocks and storage blocks.

The DTU processes all inputs into a PCM time multiplex format which biphase modulates a 2048-Hz subcarrier. This subcarrier is the output of the DTU which modulates the transmitter. The inputs to the DTU consist of digital data from the experiment packages and bi-level and analog data from the spacecraft performance measurements. The DTU receives only digital data from the experiment package since all data processing and A/D conversion is accomplished by the individual experiments (except for the radio propagation).

Signals from the command decoder select different bit rates and modes of operation via uplink transmission from the tracking stations. The range of the telemetry transmission bit rates is from 8 to 512 bps in five steps. Any one of the five bit rates may be commanded for each of four transmission formats.

The DTU also provides various timing and gating pulses to the experiments as required.

---

*Detailed information on the Pioneer VI data handling system can be found in NASA/ARC document PC-046.01.

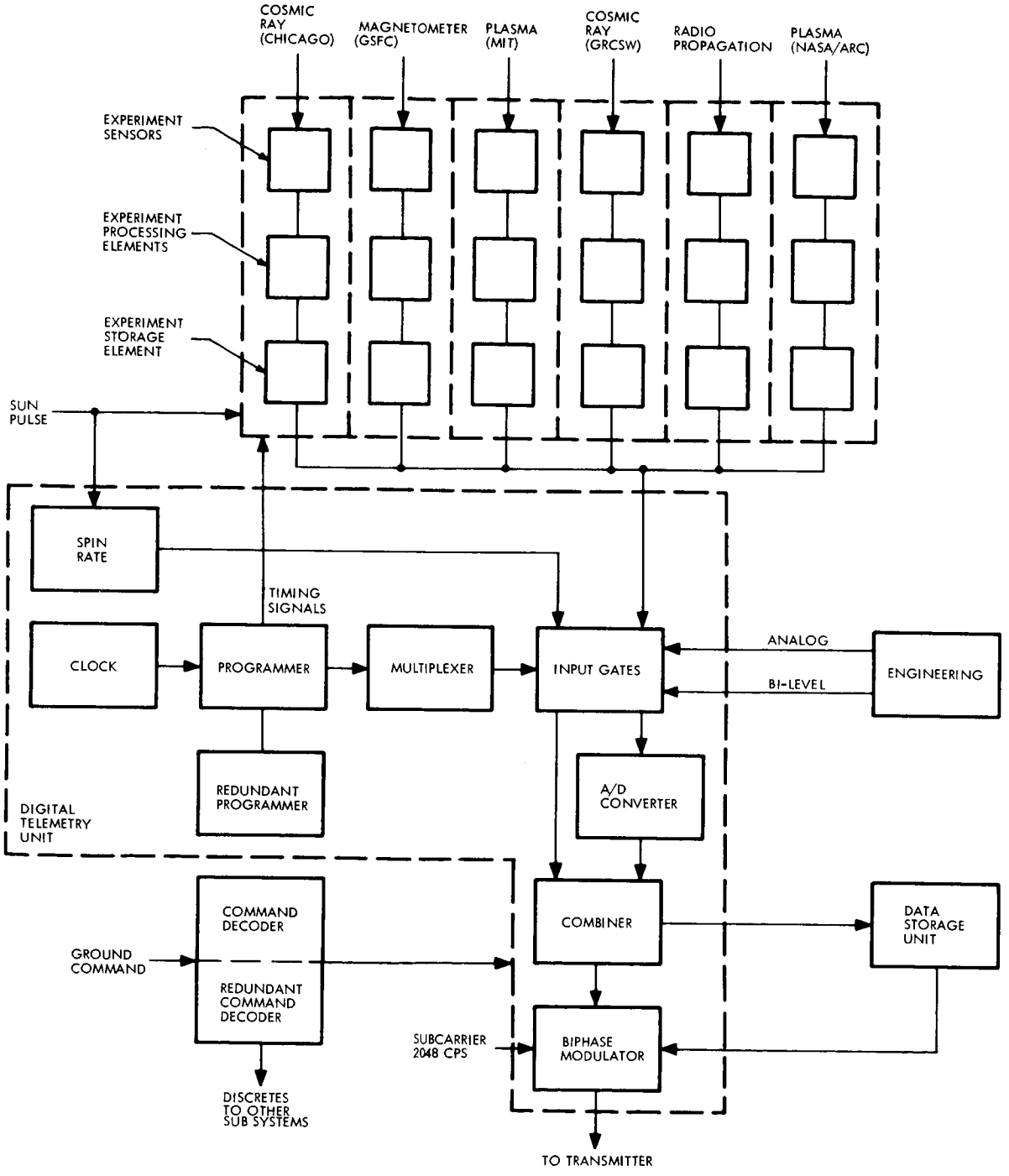


Figure 9-1. Pioneer VI Data Handling System Simplified Block Diagram

The DSU provides storage for special processing during realtime and during the time when the DSIF is not available. When transmission of stored data is commanded, the output of the DSU is read out by the DTU.

Table 9-1 summarizes the characteristics of the CDS and the Pioneer VI data handling system.

9.3 COMPARISON OF DATA TRANSMISSION EFFICIENCY

A major advantage of the CDS is its high data transmission efficiency. That is, the CDS bits transmitted are all data or convolutional coding bits. A study of the Pioneer VI transmission formats indicates that as high as 41.2 percent of the bits transmitted are noninformation bits or fill-in bits in the format (bit rate of 512 bps). This occurs because the experiments cannot sample and process data to maintain a constant data flow to the DTU. Therefore, if the DTU interrogates a particular experiment in its normal sequence and the experiment data is not available, fill-in bits are put in the frame in place of data. Hence, the earliest that the experiment data can be transmitted is the next frame. The result is that an uneven sampling of the scientific environment occurs. The CDS, on the other hand, transmits data in constant, regular intervals at all times, even during periods of noncommunication. That is, during periods of noncommunication, the CDS stores data for transmission when communication is re-established. The Pioneer VI system does have a duty cycle store mode which is used to store data during noncommunication periods. However, only 15K-bit capacity is provided which has proven to be far too small a storage capacity to provide meaningful data for a storage period of 13 hours without the use of data reduction techniques. Table 9-2 lists the fill-in bits and data bits transmitted by the Pioneer VI over a period of 1 hour. The table reflects bits transmitted in format A at a bit rate of 512 bps.

9.4 SUMMARY OF BASIC ADVANTAGES OF CDS

The advantages of the CDS are:

1) Flexible overall design
   - Universal design for many missions
   - A one time development and qualification cost

Table 9-1. Comparisons of CDS and Pioneer VI

	Pioneer VI	CDS
Number of inputs processed and transmitted	15 digitals 71 analogs 66 bi-levels 1 spin rate Additions cannot be made without redesign due to the fixed formats used.	27 pulse data, 91 analog, 17 bi-levels. Additional inputs may be accommodated by adding sampler gates and reprogramming the processor instruction memory to process additional signals. (Section 3.1)
Sampling rate (assuming 1 rps spin rate)	Max of 2.3 samples/experiment/second	64 samples/exp/sec (Section 3.3.1)
Number of formats	4 (fixed)	Programmable, typically 8 programmable, 2 fixed (Appendix B)
Format size	224 bits (fixed)	Programmable, typically 2K bits, 1K, 0.5K, and 0.25K bits (Appendix B)
Spacecraft word size	(fixed) 6 bits + one parity (7-bit word size)	Programmable, word size variable
Transmission bit rates	512 256 64 16 8	2048 1024 512 128 64 32 16 (Section 3.3.4.4)
Modes of operation	Realtime Telemetry store Duty cycle store Memory readout	Program controlled: Transmission of realtime interleaved with stored data Transmission of realtime data only Data storage operation during noncommunication periods (bulk storage)

Table 9-1 (Continued)

	Pioneer VI	CDS
		Transmission of high rate engineering data only Self test Fixed operations: Program verification Realtime format Data storage readout (Section 3.3)
Ground commands	128	128 (Section 4.4.3)
Number of experiments	6 (fixed)	8 (more can be added) (Section 2.2)
Reliability	0.68 for 6 months	0.67 for 1 year 0.54 for 1.5 years 0.43 for 2 years (excludes bulk store tape unit) (Section 6.3)
Data storage (bits)	Data handling system - 15,232 Experiments - 2292 Total - 17,524 bits	56,000, includes 2 input data buffers, 2 output buffers, and the auxiliary memory. Bulk store = 2×10^7 (Section 3.3.2.1)
Weight (lbs)	Data handling - 10.6 lbs Decoder - 6.0 lbs 6 experiments - 34.7 lbs Total - 51.3 lbs	CDS 16.5 lbs 8 experiments minus CDS processing circuits (does not include bulk store) 11.0 lbs 27.5 lbs (Section 5)
Power (watts)	Data handling - 4.4 w 6 experiments - 7.3 w Decoder - 1.4 CDU negligible Total - 13.10	CDS (total) 8.9 w 8 experiments 11.0 w Total 19.9 w (Section 5)

Table 9-1 (Continued)

	Pioneer VI	CDS
Ground stations	DSIF	DSIF
Failure modes	1 redundant programmer 1 redundant command decoder	Failure mode bypass capabilities by reprogramming CDS to bypass failed component (in processor instruction memory, IDB, processor, etc.) (Section 4.2.1)
Data transmission efficiency	As many as 40% of bits transmitted consists of fill-in bits - (format A 512 bps)	No fill-in bits required (Section 9.3)
Data processing	Fixed processing within each experiment	Programmable by central processor: Log scaling Min-max Max Averaging (Section 4.2.9)
Number of sectors sampled per spacecraft revolution	Fixed by experiments 16 max	64 sectors for each synchronous experiment (Section 3.3.1)
Number of A/D required	6 within experiments 1 in DTU	2 (1 redundant) (Section 3.3)
Bulk storage	15,232 bits (core)	2×10^7 bits (tape) (Section 4.5)
Data sampling sequence	Dependent on particular experiment data processing timing (not constant)	Constant intervals (Section 3.3.1)

Table 9-1 (Continued)

	Pioneer VI	CDS
Elapsed time counter (spacecraft clock)	Variable recycling time dependent on bit rate (8 minutes to 8 hours)	Central spacecraft clock (recycling time 194 days, 1 second resolution) elapsed time transmitted within each frame (Section 3.3.5)
Spacecraft sector generator	Generated by individual experiment package	1 central sector generator (Section 3.3.1)
Analog encoding accuracy (quantitization level)	2^6 = 64 levels	2^8 = 256 levels

Table 9-2. Data Transmission Efficiency of Pioneer VI System

	Bits/Hour		
	Data	Fill-in Bits	Total
Chicago cosmic ray (SIMPSON) η = 43.8% - 7/16 frames are data*	126,100	161,900	288,000
Magnetometer (NESS) η = 43.8% - 7/16 frames are data*	101,000	129,700	230,700
Plasma ARC (WOLFE) η = 73.8%	512,000	180,000	692,000
(GRCSW) cosmic ray η = 44.6%	25,750	31,950	57,700
Plasma mit (Bridges) η = 46.2%	79,900	93,100	173,000
Radio propagation (STANFORD) η = 87.5%	50,500	7,200	57,700
	895,250	603,850	1,499,100
2.67 frames/rev*	59.8%	41.2%	

*At 512 bps transmission bit rate, format A.

- Standard signal input interface.

2) Efficient data transmission. Since the format size is programmable and data sampling is independent of data processing, every bit within the transmission format is utilized; that is, no fill-in bits are inserted within the format.

3) Programmable data processing. The processor is programmed to optimize data processing and transmission for each particular mission. The CDS may be reprogrammed via the command link.

4) Storage of large quantities of data. A bulk storage unit is provided with capability of storing approximately 10^7 bits at each transmission bit rate.

5) Sharing of processing equipment. The CDS shares processing equipment among spacecraft experiment inputs thereby reducing circuit redundancies.

6) The processor performs data processing and data compression. The CDS processor performs data processing such as averaging, minimum-maximum, maximum, log scaling, and any other data manipulation that may be required by the experiments.

7) Self test capability. The CDS has the capability of calling upon a self test program which provides a means to evaluate CDS performance during the mission.

8) Constant known sampling of data. The CDS collects 64 samples per spacecraft revolution for every experiment.

9) The CDS has a central elapsed time clock.

10. RECOMMENDATIONS AND CONCLUSIONS

The results of the study indicate that a CDS is practical for use in a lightweight deep space probe in terms of meeting the requirements for weight, power, and flexibility.

Because of the favorable results obtained in meeting the goals of the study and the improvements it provides over the Pioneer VI type system, it is recommended that the CDS design be carried out through a detailed logic and circuit design. Specifically, it is recommended that the following areas be further studied for the detailed logic design:

1) Reduce required average power by use of power gating techniques.

2) Use one central memory system as opposed to separate memories (i.e., processor instruction memory, auxiliary memory, input data buffer, and output buffer).

3) Or, as an alternative, use CDS memories interchangeably. For example, use the auxiliary memory in place of one or both input data buffers.

4) Standardize memory sizing in terms of word length, modularization packaging, etc.

5) Extend data reduction capabilities and techniques. Increase the computational capability of the processor to perform more extensive data reduction (i.e., statistical analysis).

6) Use the processor to perform convolutional coding.

7) Use an NDRO processor instruction memory.

APPENDIX A
GLOSSARY OF TERMS

APPENDIX A
GLOSSARY OF TERMS

MAG	Magnetometer experiment
VLF	Very low frequency experiment
C. COS	Chicago cosmic ray experiment
W. COS	Webber's cosmic ray experiment
PLA	Plasma experiment
CH	Channel
A	Analog
PHA	Pulse height analysis
D_1	Chicago cosmic ray experiment detector outputs
D_2	Chicago cosmic ray experiment detector outputs
D_3	Chicago cosmic ray experiment detector outputs
D_4	Chicago cosmic ray experiment detector outputs
C	Webber's cosmic ray experiment detectors
B_1	Webber's cosmic ray experiment detectors
B_2	Webber's cosmic ray experiment detectors
B_3	Webber's cosmic ray experiment detectors
B_i	Bi-level data
D_i	Digital data
RAP	Radio propagation experiment
TEL 3.1	Webber's cosmic ray experiment telescope outputs
TEL 3.2	Webber's cosmic ray experiment telescope outputs
TEL 3.3	Webber's cosmic ray experiment telescope outputs
TEL 4.1 ↕ TEL 4.9	Webber's cosmic ray experiment telescope outputs
ET_1	Elapsed time (least significant 8 bits, 1-sec resolution)
ET_2	Elapsed time (second most significant 8 bits)
ET_3	Elapsed time (most significant 8 bits)
MIC	Micrometeroid experiment
TOF	Micrometeroid time of flight
ω_t	Spacecraft spin rate
OP	Processor instruction operation code

$A_1 - A_6$	Sampler fixed programmer outputs
DSIF	Deep space instrumentation facility
CDS	Central data system
OB	Output buffer
A/D	Analog to digital
M_1	Marker signals from output buffer
M_2	Marker signals from output buffer
E_X	X electric field VLF experiment
E_Y	Y electric field VLF experiment
E_Z	Z electric field VLF experiment
A	A event micrometeroid experiment
B	B event micrometeroid experiment
M	M event micrometeroid experiment
S_1	Radio propagation phase path counter
S_2	Radio propagation 423.6-MHz carrier amplitude
S_3	Radio propagation 10-kHz modulation phase
S_4	Radio propagation 49.8-MHz carrier loop stress signal
S_5	Radio propagation 49.8-MHz carrier amplitude
S_6	Radio propagation 49.8-MHz carrier amplitude (lunar occultations mode)
NDRO	Nondestructive readout
AU	Astronomical unit
MWI	Memory write in
ϕ	Phase
AR1	Accumulator register
AR2	Auxiliary register
AR3	Auxiliary register
A1	8 most significant bits of AR1
A2	8 next most significant bits of AR1
A3	8 least significant bits of AR1
MSB	Most significant bit
LSB	Least significant bit
MAR(I)	Memory address register of input data buffer
MAR(P)	Memory address register of processor instruction memory
MAR(A)	Memory address register of auxiliary memory

MAR(O)	Memory address register of output buffer
MR(I)	Input-output register of input data buffer
MR(P)	Input-output register of processor instruction memory
MR(A)	Input-output register of auxiliary memory
MR(O)	Input-output register of output buffer
R1	Auxiliary memory increment register
R2	Auxiliary memory stop address register
R3	Stop sector address register
R4	Successive sector address register
R5	Sector increment register
R6	End of frame register
Cp	Clock pulse
W_1-W_{64}	Main frame words
G_1	Input data sampler gate
T_1-T_{64}	Timing pulses
P	Parity
MRO	Memory readout

APPENDIX B
TRANSMISSION FORMATS

APPENDIX B
TRANSMISSION FORMATS

Appendix B contains realtime and bulk store formats described in Figure B-1, high rate engineering format described in Figure B-2, and fixed realtime transmission format described in Figure B-3.

1) Low Rate Data

Same as programmable mode (see Section 3.1.3.1)		64 bits

2) Subcomm

Same as programmable mode		8 bits

3) VLF

X_c, Y_c, Z_c (uncorrelated data)		24 bits

4) RAP

Sample S_5 two times		16 bits

5) Plasma

Sample flux four times		32 bits

6) C. COS

DHA $D_1 D_2 \overline{D}_3 \overline{D}_4$	16 bits	40 bits
DHA $D_1 D_2 D_3 \overline{D}_4$		
Pulse data	24 bits	

7) W. COS

PHA (c)		
$(B_1 B_2 B_3)_{min}$	24 bits	48 bits
$(B_3 - B_1)$		
Pulse data	24 bits	

8) MAG

A/D X, Y, and Z axis		24 bits

	FRAME IDENTIFICATION				HIGH RATE DATA					
FORMATS	FRAME SYNC	SUB-COMM ID	FORMAT ID	BIT RATE	MAG*	C COS	PIA	RAP	VLF	W COS
1	24 Bits	6	4	3	192	312	125	112	24	213
2	17	6	4	3	96	164	41	24	18	156
3	11	6	4	3	24	65	20	24	18	78

Note: Format 1) TYPICAL FORMAT FOR 2048 AND 1024 BPS TRANS RATE

Format 2) TYPICAL FORMAT FOR 512, 256, AND 128 BPS TRANS RATE

Format 3) TYPICAL FORMAT FOR 64, 32, 16 BPS TRANS RATE

*REFER TO APPENDIX A FOR DEFINITIONS OF ABBREVIATIONS

Δ LOW RATE DATA AND SUBCOMM REMAIN THE SAME FOR ALL BIT RATES

LOW RATE DATA 64 BITS

START TIME OF SAMPLING CYCLE (ET_1) 8

START SECTOR ADDRESS OF SAMPLING CYCLE 8

PLASMA VOLTAGE, CHANNEL ID 8

EXP NORMAL CALIBRATE STATUS, MIC RAP, VLF & FAN ID 6

Bits 30

RAP

S_1 10

S_2 6

S_3 6

S_4 6

S_6 6

Bits 34

VLF*

X_c 6

Y_c 6

Z_c 6

M_1 6

M_2 6

Filter 4

Bits 34

MIC*

A, B MATRIX . . 8

ENERGY, MOM . 8

TOF 4

SUN ASPECT . . 7

A, B, M . . . . . 3 EVENTS

MIC ELAPSED TIME. 4

Bits 34

FAN

BLOCK OF 34 BITS

SUB COMM Δ (8/Bits/ Frame)

ENG DATA-48 words

BI-LEVEL 2 words

INCLUDES

ET_2 (one word)

ωt (one word)

MAG (one word)

Range

Offset

Orientation

ET_3

54 8-BIT WORDS

Figure B-1. Realtime and Bulk Store Formats

FRAME SYNC	BIT RATE ID	FORMAT ID	DATA 64 8-BIT WORDS
17	3	4	512 BITS

NOTE

1) THIS FORMAT MAY BE USED AT ALL EIGHT BIT RATES

2) ELAPSED TIME INCLUDED WITHIN THE 64 EIGHT BIT GROUPS

Figure B-2. High Rate Engineering Format

FRAME SYNC	FRAME SYNC	SUBCOMM ID	FORMAT AND BIT RATE ID
MAG	MAG	MAG	C COS
C COS	C COS	C COS	C COS
W COS	W COS	W COS	W COS
W COS	W COS	PLA	PLA
PLA	PLA	RAP	RAP
VLF	VLF	VLF	LOW RATE DATA
LOW RATE DATA	LOW RATE DATA	LOW RATE DATA	LOW RATE DATA
LOW RATE DATA	LOW RATE DATA	LOW RATE DATA	SUB COMM

NOTE:

1) THIS FORMAT USED FOR 256 AND 64 AND 16 BPS TRANSMISSION RATES
2) EACH SQUARE CONTAINS 8 BITS
3) REFER TO APPENDIX A FOR DEFINITION OF ABBREVIATIONS
4) REFER TO FOLLOWING PAGE FOR EXPLANATION OF FORMAT

Figure B-3. Fixed Realtime Transmission Format 286-Bit Frame

9) Frame Synchronization

Two 8-bit words are used for frame sync	16 bits

10) Subcomm ID

One word is used to identify which of the 64 subcomm words is transmitted (2 spare bits)	8 bits

11) Format and Bit Rate ID

Four bits used for format ID and two 3-bits used for bit rate ID (2 spare bits)	8 bits
TOTAL	288 bits

APPENDIX C
DATA FLOW DIAGRAM

APPENDIX C
DATA FLOW DIAGRAM

Figure C-1 is a general data flow diagram for typical experiments. The plasma experiment is used to illustrate the data flow of Figure C-1 and is described below.

Spacecraft Revolution No. 1 (data sampling)

1) Sample one selected data point each spacecraft sector (64 times).

2) Encode the plasma data channels into 8 bits by the use of the A/D.

3) Store 64 samples into IDB word locations (11 through 14) corresponding to the sector addresses from which the samples were taken.

Spacecraft Revolution No. 2 (data processing)

1) Select all 64 samples from IDB using the fetch sequential subroutine. See Section 4.2.3.2 for explanation of fetch subroutine.

2) Calculate the maximum value of flux and identify with sector address. See Section 4.2.9 for explanation of maximum value calculation routine.

3) Every fourth sample is formatted and stored within the output buffer followed by the maximum flux value. See Section 4.2.6 for explanation of the "put" command sequence.

Spacecraft Revolution No. 3 (data transmission)

1) Transmit data frame stored within output buffer which contains 125 bits of plasma data.

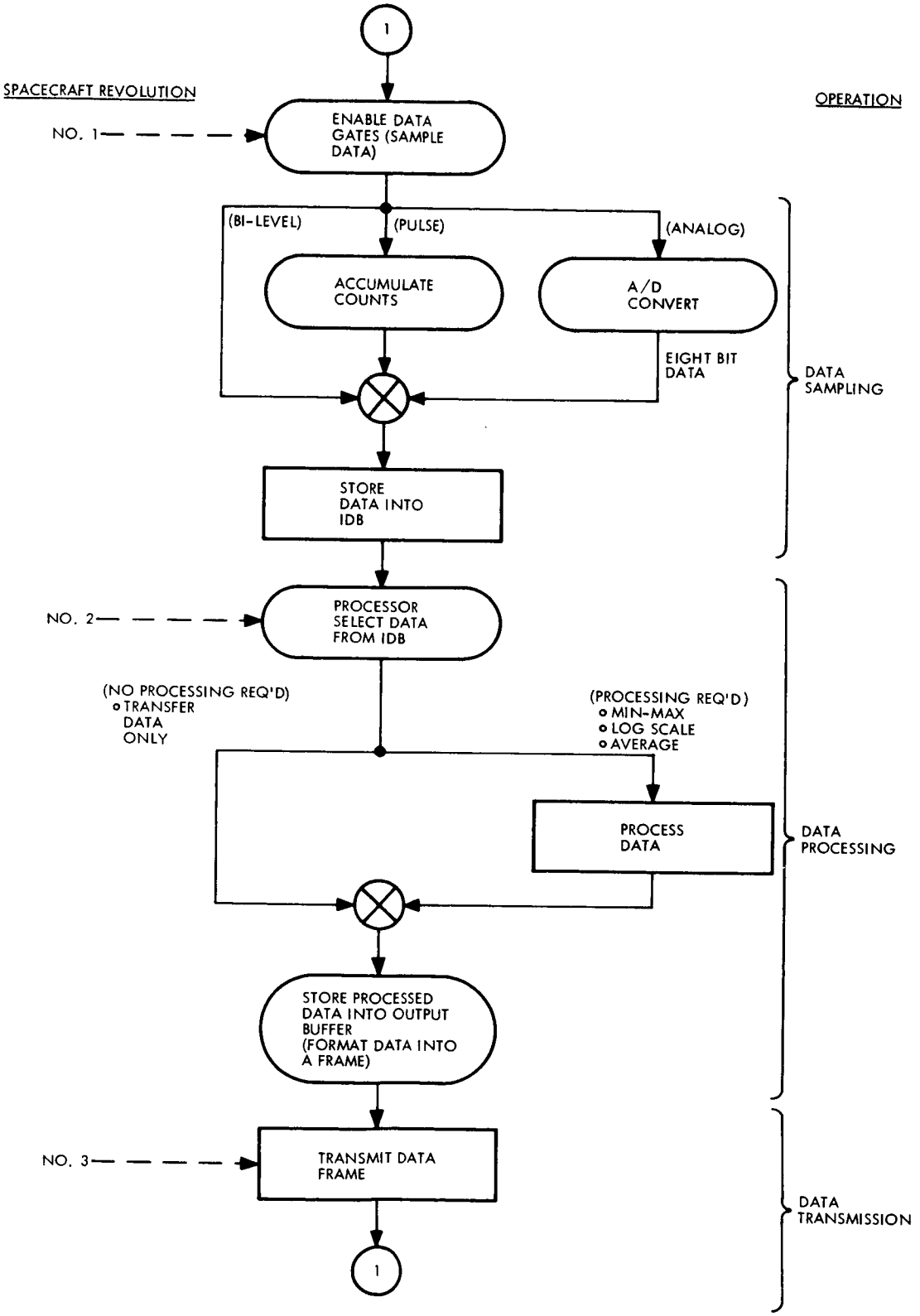
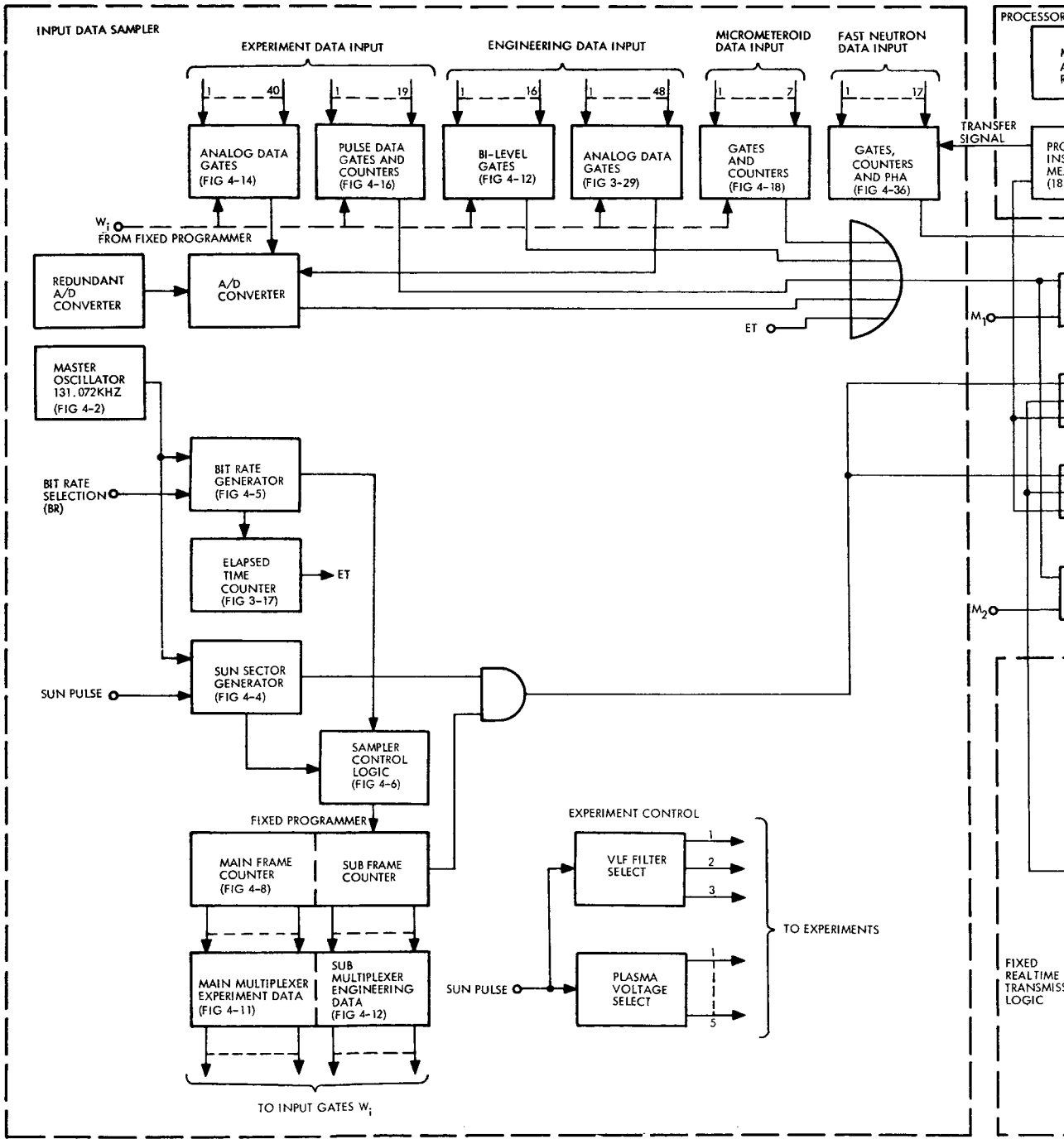


Figure C-1. Data Transmission Data Flow Diagram

APPENDIX D
DETAIL CDS BLOCK DIAGRAM


INPUT DATA SAMPLER
EXPERIMENT DATA INPUT
ENGINEERING DATA INPUT
MICROMETEROID DATA INPUT
FAST NEUTRON DATA INPUT
1 40
1 19
1 16
1 48
1 7
1 17
ANALOG DATA GATES (FIG 4-14)
PULSE DATA GATES AND COUNTERS (FIG 4-16)
BI-LEVEL GATES (FIG 4-12)
ANALOG DATA GATES (FIG 3-29)
GATES AND COUNTERS (FIG 4-18)
GATES, COUNTERS AND PHA (FIG 4-36)
TRANSFER SIGNAL
W_i
FROM FIXED PROGRAMMER
REDUNDANT A/D CONVERTER
A/D CONVERTER
ET
M_1
MASTER OSCILLATOR 131.072KHZ (FIG 4-2)
BIT RATE SELECTION (BR)
BIT RATE GENERATOR (FIG 4-5)
ELAPSED TIME COUNTER (FIG 3-17)
ET
M_2
SUN PULSE
SUN SECTOR GENERATOR (FIG 4-4)
SAMPLER CONTROL LOGIC (FIG 4-6)
FIXED PROGRAMMER
MAIN FRAME COUNTER (FIG 4-8)
SUB FRAME COUNTER
EXPERIMENT CONTROL
VLF FILTER SELECT
1
2
3
TO EXPERIMENTS
MAIN MULTIPLEXER EXPERIMENT DATA (FIG 4-11)
SUB MULTIPLEXER ENGINEERING DATA (FIG 4-12)
SUN PULSE
PLASMA VOLTAGE SELECT
1
5
TO INPUT GATES W_i
PROCESSOR
FIXED REALTIME TRANSMISS LOGIC

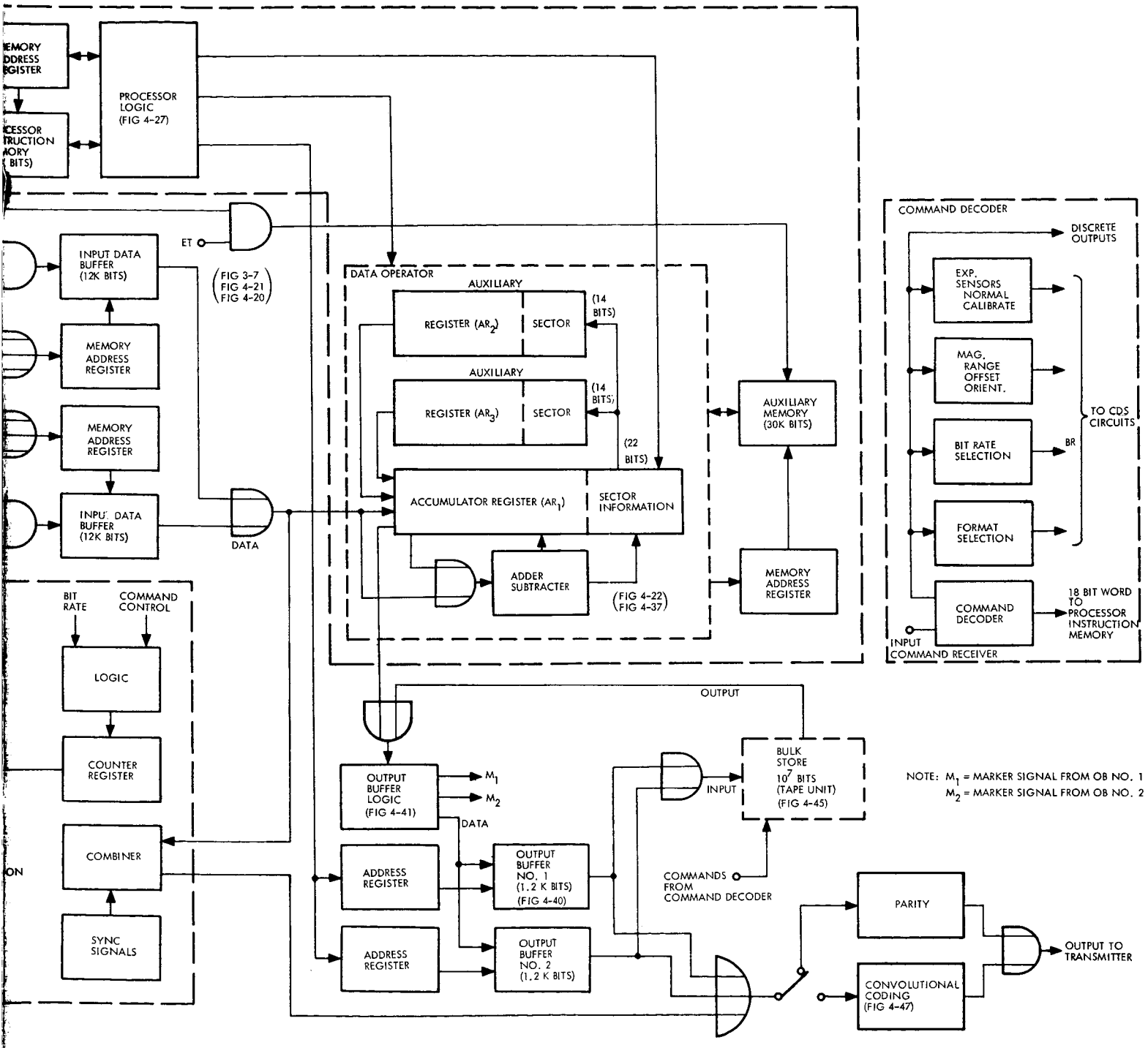


Figure D-1. Detailed CDS Block Diagram